CELTIC Project Number:	**C2012/2-5**
Project Title:	SDN Concept in Generalized Mobile Network Architectures SIGMONA
Confidentiality:	PU / RE / CO[1]

Document Identifier:	D3.2
Document Title:	**Software-defined mobile network-ready advanced traffic and resource management**
Authors:	Aydın Ulaş, Onur Koyuncu
Participants:	See authors section
Work Package:	WP3
Version:	Under BSCW version control
Date of last changes:	15.03.2016
File Name:	D3.2.pdf

Abstract:	This document contains Software-defined mobile network-ready advanced traffic and resource management in SDMNs. The document explains for each use-case the proposed solution, architecture and expected benefits and possible integration scenarios.

Keywords:	solutions, traffic management, software-defined mobile networks, results

Disclaimer:

Document History:	
25.06.2015	Document created
15.03.2016	Final draft

[1] Dissemination level:
PU = Public
RE = Distribution to a group specified by the consortium
CO = Confidential, only allowed for members of the consortium

Table of Contents

Authors

Partner	Name	Phone / Fax / e-mail
Budapest University of Techn. and Econ., Mobile Innovation Centre		
	Zoltán Faigl	
		Phone: +36 1 463 2500
		e-mail: zfaigl@mik.bme.hu
	László Bokor	
		Phone: +36 1 463 2048
		e-mail: bokorl@hit.bme.hu
Technical University of Chemnitz		
	Thomas Bauschert	
		Phone:
		e-mail: thomas.bauschert@etit.tu-chemnitz.de
	Marcus Eckert	
		Phone:
		e-mail: marcus.eckert@etit.tu-chemnitz.de
NEXTEL		
	Oscar López	
		Phone:
		e-mail: olopez@nextel.es
	Mikel Uriarte Itzazelaia	
		Phone:
		e-mail: jnebrera@eneotecnologia.com
ENEO		
	Jaime Nebrera	
		Phone:
		e-mail: jnebrera@eneotecnologia.com
Innovalia Association		
	Daniel Alcaraz Real-Arce	
		Phone:
		e-mail: dalcaraz@innovalia.org
Turk Telekom Argela		
	Aydın Ulaş	
		Phone:
		e-mail: aydin.ulas@argela.com.tr
	Onur Koyuncu	
		Phone:
		e-mail:onur.koyuncu@argela.com.tr

Ericsson Turkey		
	Hasan Anıl Akyıldız	
		Phone:
		e-mail: hasan.anil.akyildiz@ericsson.com
	Ece Saygun	
		Phone:
		e-mail: ece.saygun@ericsson.com

CEA		
	Mohamed LABRAOUI	
		Phone:
		e-mail: mohamed.labraoui@cea.fr
	Michael BOC	
		Phone:
		e-mail: Michael.boc@cea.fr

Executive Summary

This document describes solutions for optimized resource and traffic management in macroscopic and microscopic level for software-defined mobile networks (SDMN).

The solutions cover the following topics:

Resource management (RM):

- Resource availability in SDMNs

Macroscopic-level traffic management:

- Coordinated traffic and resource management and efficient routing in SDMNs
- Application-Layer Traffic Optimization techniques in SDMNs
- Joint traffic and cloud resource management for service chaining in SDMNs
- SDN-controlled device-to-device communications

Microscopic-level traffic management:

- DiffServ QoS architecture in SDMNs
- QoE/QoS enforcement in SDMNs
- Quality of Service Control and Resource Prioritization with SDN

List of abbreviations

3GPP	Third Generation Partnership Project
AF	Application Function (in PCC architecture), Assured Forwarding (In DiffServ concept)
ALTO	Application-Layer Traffic Optimization
API	Application Programming Interface
BBERF	Bearer Binding and Event Reporting Function
CDN	Content Distribution Network
DHCP	Dynamic Host Configuration Protocol
DiffServ / DS	Differentiated Services
DPI	Deep Packet Inspection
DSCP	Differentiated Services Code Point
GGSN	Gateway GPRS Support Node (GGSN)
GPRS	General Packet Radio Service
GTP	GPRS Tunneling Protocol
IaaS	Infrastructure-as-a-Service
IETF	Internet Engineering Task Force
IntServ	Integrated Services
IMS	IP Multimedia Subsystem
IND	Infrastructure Network Domains
IPv4 / Ipv6	Internet Protocol version 4 / version 6
ISAAR	Internet Service quality Assessment and Automatic Reaction
ISG	Industry Specification Group
MME	Mobility Management Entity
MNO	Mobile Network Operator
NFV	Network Function Virtualization
NMS	Network Management System
OF	OpenFlow
OFS	OpenFlow Switch
ONF	Open Network Foundation
P2P	Peert-to-Peer
PCC	Policy Control and Charging
PCEF	Policy Control Enforcement Function
PCRF	Policy and Charging Rules Function
PDN	Packet Data Network
P-GW	PDN Gateway
PHB	Per Hop Behaviour
PID	Provider-defined Identifier
QCI	QoS Class Identifier
QENF	Quality Enforcement
QMON	Quality Monitoring
QRULE	Quality Rules Entity
QoE	Quality of Experience
QoS	Quality of service
RM	Resource Management
SCO	Service Chaining Orchestrator
SDN	Software-Defined Network
SDMN	Software-Defined Mobile Network
SDNC	SDN Controller Platform

SDP	Session Description Protocol
SIP	Session Initiation Protocol
SLA	Service-Level Agreement
SPR	Subscription Profile Repository
TE	Traffic Engineering
TM	Traffic Management
UE	User Equipment
VIM	Virtual Infrastructure Manager
VMNO	Virtual Mobile Network Operator
VM	Virtual Machine
VNF	Virtual Network Function

1. Introduction

Traffic and resource management in software-defined mobile networks (SDMN) can be implemented more efficiently and intelligently compared to the conventional approaches to exploit the major advantages of the SDMN architecture. More specifically, SDMN provides centralized visibility including global network information (e.g., network resource limitations or dynamically changing status of the network) and global application information (e.g., QoS/QoE requirements). Therefore, SDMN provides end-to-end communication across multiple distinct technologies, such as 3G, 4G, Wi-Fi, etc. with solutions on traffic and resource optimization for dynamic environments.

Traffic and resource management is of critical importance to the evolution and success of SDMNs to optimize the performance of the network by dynamically analyzing, predicting, and regulating the behavior of the data. SDMN enables the optimized traffic and resource management based on the individual service needs. This is especially important in the mobile environment where end users constantly change their location, bandwidth demands vary widely depending on the type of content being sent and wireless coverage is not uniform.

This deliverable describes preliminary technological solutions for optimized resource and traffic management in SDMNs, proposed in the framework of SIGMONA project. Section 2 summarizes analysis of the challenges and resource directions for resource management, macroscopic and microscopic traffic management separately. Section 3 presents the proposed solutions and architectures including main benefits, results and future works. Section 4 addresses the architectural modifications with detailed interface descriptions, integration of technologies including possible performance improvements.

2. Challenges and Research Objectives

This section summarizes the research objectives and main research challenges of traffic management topics. The following topics are addressed in the document:

Resource management:

- Network resource availability awareness

Macroscopic Traffic management:

- Coordinated traffic and resource management and efficient routing in SDMNs
- SDN-based Application-Layer Traffic Optimization
- Joint traffic and cloud resource management for service chaining in SDMNs
- SDN-controlled device-to-device communications

Microscopic Traffic management:

- Diffserv QoS Architecture for SDN transport with policy control using PCRF
- QoE/QoS enforcement in SDMNs

2.1 Resource Management

2.1.1 Network resource availability awareness

Network resource availability awareness will provide better modularity for scalability and interoperability with different solutions when using different SDN implementations, as a result legacy physical managers could aggregate and include their information in a unique platform and facilitating further managing actions over network resources.

2.1.1.1 Analysis of the challenges

Network resource availability may provide solution of heterogenity of network resources and dynamicity of them in SDN and SDMN environments.

2.1.1.2 Research directions

Information regarding virtual and physical networks will be collected for status monitoring and to be presented in a centralized graphical user interface. This will consolidate global network view where different SDN controller based managers will provide information.

2.2 Macroscopic Traffic Management

2.2.1 Coordinated traffic and resource management and efficient routing in SDMNs

Coordinated traffic management and efficient routing algorithms for software defined mobile networking will collect up-to-date global network state information such as bandwidth, the number of available port and reduce power consumption while satisfying users' Quality of Service (QoS) constraints such as throughout and/or delay. The power consumption can be determined as the number of active switches and/or the number of links in the network.

2.2.1.1 Analysis of the challenges

The challenges are to design energy efficient low complexity traffic management and routing algorithm while satisfying given constraints for both low and high traffic load cases.

2.2.1.2 Research directions

- The traffic management and efficient routing will be developed to reduce power consumption by considering flow's rate requirements under the assumption that the network load is low.
- The traffic management and efficient routing will be designed to reduce power consumption by considering flow's rate requirements and the number of available ports under the assumption that the network load is high, medium and low.

2.2.2 Application-layer traffic optimization in SDNs

By the integration of ALTO network information service into SDNs, ALTO becomes transparent for the UE or the service claimant entity (no deployment cost in the UE). Due to ALTO information, the ALTO client in the SDN controller can overwrite the initial peer selection decision of the service claimant entity (e.g. UE). Any flow can be dynamically selected for getting ALTO guidance and SDN providses built-in redirection mechanisms.

2.2.2.1 Analysis of the challenges

For ALTO-SDN integration the following challenges should be tackled. We need to develop an ALTO client integrated as an application module in the SDN controller. We need to develop an ALTO server and realize the ALTO Client-to-Server interface. The API functions must follow the ALTO protocol specification [1].

We mainly target ALTO-aware HTTP-based services. We must specify and develop the flow redirection mechanism in the SDN environment, using HTTP Redirect server.

ALTO-SDN may run over any routing protocol. We need some type of routing in and beyond SDN-based network areas.

We need to specify ALTO Network-to-Server APIs, automatic network map provision, cost map provision, and merging of map information.

Selection of ranking aggregation techniques for ALTO guidance based on multiple cost types is an open area. ALTO protocol does not specify the ranking algorithm.

2.2.2.2 Research directions

Regarding the ALTO-SDN topic, we choose to develop a proof-of-concept implementation to demonstrate the operation and benefits ALTO-SDN integration.

2.2.3 Joint traffic and cloud resource management for service chaining in SDMNs

2.2.3.1 Analysis of the challenges

Creation of high performance service-chaining applications contributes to the fulfillment of traffic demands and higher quality expectations from customers, while reducing capital and operational expenses associated with their networks. To achieve this, traffic should be processed only in the necessary middleboxes and should refrain from processing in unnecessary middleboxes. Detailed identification of each flow, and establishing an appropriate dynamic service-chain for that flow is the main problem. In this project,

Ericsson will work on developing an OpenFlow-based SDMN architecture where the SDMN orchestration controller will control not only the flow forwarding hardware, but also the middleboxes that are geographically distributed in the Network-Enabled Cloud by using the traffic and cloud resource management methods.

2.2.3.2 Research directions

Ericsson will be following the following research path:

- Develop methodology to periodically gather service-based server load and path congestion information from Network Management System or other equivalent systems.
- Develop a routing mechanism, which utilize the above methodology; implement this as a module over OpenDaylight SDN Controller.
    - Implement this mechanism over Mininet environment
    - Implement this over actual OF switch and OpenDaylight Controller with selected applications on the forwarding path.
- Develop a mechanism for optimizing the service chaining for the case of some services being provided over the cloud, especially for Network as a Service (NaaS) case.

2.2.4 SDN-controlled device-to-device communications

This research topic addresses the opportunity of offloading the Radio Access Network by the use of IP Wireless Mesh Network (e.g. WiFi). Here we consider SDN control of IP communications in the edge networks, meaning that smartphones are SDN capable. It is commonly assumed that such a target would be handled by end-terminals themselves as a completely distributed system without mobile network supervision. This research topic investigate wether the mobile network operator can keep control of these communications to redirect traffic to a different access network (e.g., fixed).

2.2.4.1: Analysis of the challenges

- Specify solutions to allow coordinated and efficient IP Wireless Mesh communications between end-terminals (such as smartphones) by a supervisor.
- Define a software architecture to provide efficient support of SDN on smartphones and lightweight mobile devices.
- Specify and implement topology control algorithm for such kind of network.

2.2.4.2: Research directions

Aiming to address the evoked challenges above, CEA established the following research plan:

- Define a strategy to monitor in real time the workload of the radio access network:
    - Choosing the pertinent metrics to take in account.
    - Develop techniques to calculate the considered metrics.
    - Define criterions and thresholds in order to declare the state of congestion.
- Develop methodology to detect a potential mesh network allowing to get better performances.
    - Develop techniques to estimate the potential performances of both cellular and mesh networks.
    - Define criterions to choose mobiles being part of the potential mesh network.
- Develop routing algorithm enabling to redirect traffic to a different access network.
- Investigate through simulations, whether the use of the SDN controller to assist a mesh network allows overcoming the limitations of the conventional "self-organized" device to device networks.

2.3 Microscopic Traffic Management

2.3.1 DiffServ QoS in SDN-based transport

DiffServ QoS architecture can provide soft QoS guarantees on top of different Layer-2 technologies in a scalable manner. Our research focuses on the discovery of QoS capabilities of SDN-transport (traffic classification, shaping, policing, dropping), in order to provide QoS guarantees in L3 network forwarding paths of MNOs and VMNOs. Our objective is to reuse legacy PCRF for dynamic policy control rule provisioning in SDN-based, virtual or physical, IP-level transport network segments.

2.3.1.1 Analysis of the challenges

DiffServ QoS may provide a scalable QoS enforcement service providing soft QoS guarantees. It may enable both flow separation and sharing of supplementary link capacities. Provision of DiffServ QoS services is possible with multi-level priority queues.

Current ONF specifications (OpenFlow, OF-config) do not support multi-level priority queues, only, one-level, flat priority queues using a simplified version of hierarchical token bucket filter and hierarchical fair-service queues.

QoS should be guaranteed on virtual mobile network level, with enough granularity. Traffic aggregates mapped to different QoS classes may represent e.g., different service classes for the MNO, entire traffic of a mobile network slice, or service classes per mobile network slices. We will investigate with measurements the traffic management capabilities provided by different multi-level priority queues. It is challenging to define the appropriate level of granularity.

Interaction with legacy PCRF and the setting of policy control rules for the Diffserv QoS classes is also an important challenge.

2.3.1.2 Research directions

In regards of DiffServ QoS enforcement, we plan to study the feasibility of integration of dynamic policy control in SDN. This needs the analysis of the Gx and Rx interfaces of standard PCRF, and the investigation of REST APIs provided by SDN controllers. Based on the results of the investigation we can propose possible ways for dynamic policy control in SDNs.

2.3.2 QoE/QoS enforcement in SDMNs

In order to achieve acceptable service quality, the broad spectrum of mobile Internet services requires differentiated handling and forwarding of the respective traffic flows. The ISAAR (Internet Service quality Assessment and Automatic Reaction) framework developed by TU Chemnitz augments existing quality of service functions in mobile networks by a flow based and network centric quality of experience (QoE) monitoring and enforcement. It is meant to be an all in one solution for service quality monitoring and QoE enforcement within an operator network. It consist of three functional blocks: quality monitoring (QMON), quality rules entity (QRULE) and quality enforcement (QENF). The task of QMON is flow detection and assessment as well as QoE estimation. Within QRULE enforcement policy rules for each observed flow are created and QENF performs the flow manipulation according to these rules. The architecture and the operation of the original ISAAR framework without SDN and NFV support are described in detail in the ISAAR-paper [2].

The new version of ISAAR will make use of the SDN and NFV concept to enhance its flexibility. SDN will be applied for traffic identification as well as for steering the traffic of interest to a monitoring node. NFV will be used to virtualize single functions of ISAAR like monitoring or enforcement and to dynamically place these functions at different network and cloud locations.

2.3.3 Quality of Service Control and Resource Prioritization with SDN

We aim to analyze how Software Defined Networking (SDN) can help to ensure a high-quality uninterrupted Voice over Internet Protocol (VoIP) service accompanied by video for prioritized users and entities in a network congested with background traffic.

2.3.3.1 Analysis of the challenges

Guaranteeing a level of Quality of Service (QoS) and resource prioritization are key concepts for future networks (e.g., 5G, NGSON) and may be required in an emergency situation such as an international disaster recovery operation. Providing QoS in such scenarios with the help of SDN is promisingly feasible, via limiting bandwidth, assigning flows to different queues and/or adapting routing decisions based on network conditions.

The new trend in both cellular and public networks is pushing Future Internet and future networks to guarantee a predefined level of Quality of Service to their users or the devices/machines that are attached on. This is simply because the future networks are not being designed to depend on the infrastructure; quite the contrary, they should be capable of instant service and network creation via the virtualization flexibility provided by SDN. Some of the prominent scenarios include emergencies such as an international disaster recovery operation, where a first responder team should be able to communicate with each other and probably with their headquarters with minimal distortion. Other public safety applications and medical applications also fall in this category. Managing the enterprise cloud with charging policies based on provided QoS and video-to-video live chat are other applications that may be listed to emphasize the importance of QoS control and resource prioritization for upcoming networks.

2.3.3.2 Research directions

We aim to analyze how SDN can help to ensure a high-quality uninterrupted VoIP service accompanied by video in an emergency. On the other hand, there will also be regular users trying to communicate with one another in a network congested with background traffic, which is created by sources other than the VoIP service. These users should also receive some QoS, although less than premium, so that the impact of background traffic may be limited.

3. Proposed Technologies for RM and TM in SDN

3.1 Resource Management

3.1.1 Resource availability in SDMNs

3.1.1.1 Definition

Network resource availability awareness will be the foundation for resource management process, service provisioning and optimization traffic management, as well as security protection and monitoring process.Both physical and virtualized resources operators require to have updated information on available inventory, topology, capacity, resilience and optionally security assurance assessment.

Both physical and virtualized resources operators require to have updated information on available inventory, topology, capacity, resilience and optionally security assurance assessment.

3.1.1.2 Proposed solution and architecture

3.1.1.2.1 Description

Information regarding virtual and physical networks will be collected for status monitoring and to be presented in a graphical user interface.

This will consolidate global network view where different SDN controller based managers, i.e. OpenDaylight, NOX, FloodLight, Ofnic, or others, running over different operating systems and over different IaaS platforms, OpenStack, FI-WARE IaaS, could provide information to a central user interface, showing network status.

3.1.1.2.2 Architecture

The proposed architecture (Figure 1) will require the appropriate functions in SDN controller and SDN switches.

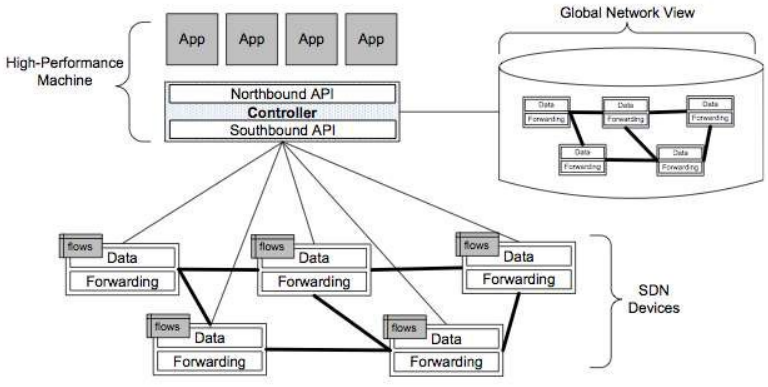


Figure 1 – Consolidating Network view with heterogeneous network resources and SDN controllers

3.1.1.2.3 Expected benefits and preliminary results

Network resource availability notably will provide better modularity for scalability and interoperability of different solutions when using different vendors or solutions in SDN; as a result, legacy physical managers could aggregate and include their information in a unique platform and facilitating further managing actions over network resources.

3.2 Macroscopic Traffic Management

3.2.1 Coordinated traffic and resource management and efficient routing in SDMNs

The joint routing and resource allocation optimization problem is finding the best path by minimizing or maximizing the cost function according to given constraints. It can be designed for different cost functions such as maximize the rate, maximize the fairness, minimize the energy, and minimize the delay.

3.2.1.1 Definition

For SDMNs, the main resources are bandwidth and flow capacity. The bandwidth refers to how many data packets a network transmits over a period. This ability refers to the forwarding capability of the network. More bandwidth means higher transmission rate and thus lower transmission delay. In this section, we define the joint routing and resource allocation optimization problem as finding the best path satisfying the given bandwidth constraints by minimizing power consumption according to given constraints.

3.2.1.2 Background

We have examined different optimization problems for joint routing and/or resource allocation in the literature for SDN considering different cost functions and contraints;

In [3], the aim is to maximize the rate of a control application based on fairness allocation of link bandwidth.

$$\max \sum_{a:a\epsilon A} ln\omega_a y_a, \forall a \in A, \quad \textbf{(3.1)}$$

where ω_a is the price of bandwidth which is paid by a control application and y_a is the sum of flow rates over the link, A is the control application set. The constraints are:

$$\sum_{p:p\in P(a)} x_{ap} = y_a, \forall a \in A \quad \textbf{(3.2)}$$

where x_{ap} is individual flow rate and P is the set of paths and $P(a)$ denotes the set of paths of a given application.

$$\sum_{p:p\in P(l)} x_{ap} \leq C_l, \forall l \in L, \forall a \in A, \quad \textbf{(3.3)}$$

where L is the set of links, P(l) denotes the set of paths crossing a given link, and C_l denotes the bandwidth of link l.

$$x_{ap} \geq 0, y_a \geq 0, a \in A, p \in P \quad \textbf{(3.4)}$$

The rate of a control application 'a' satisfies $y_a = \sum_{p:p\in P(a)} x_{ap}, \forall a \in A$ .The sum of flows over link l should not exceed the capacity of the link, i.e., $\sum_{p:p\in P(l)} x_{ap} \leq C_l, \forall a \in A$. There exists a unique optimum for the rate vector y_a for each control application because the objective function (3.1) is a strictly convex function of y_a, but there may be many corresponding values of the flow rate x_{ap} satisfying the relations (3.2) and (3.3).

In [4], the aim is to minimize the cost function subject to delay. Let $R(s, t)$ be the set of all routes where s is the source and t is the destination node. For any route $r \epsilon R(s, t)$, we define cost $f_C(r)$ and (worst case) delay $f_D(r)$ measures as,

$$f_C(r) = \sum_{l\in r} c_l \quad f_D(r) = \sum_{l\in r} d_l ,$$

where $l = (u, v) \in L$ (where L is the set of links) represents a link between the nodes u and v and d_l denotes the delay variation (jitter) between node u and v and it is measured as the first derivative (rate of change) of the delay. c_l is defined as a cost metric by taking into account packet loss and delay variation as follows:

$$c_l = (1 - \beta)d_l + \beta p_l \; for \; 0 \leq \beta \leq 1, \forall l \in L,$$

where p_l denotes the packet loss measure for the traffic on link (u,v) and β is the scale factor which will be explained later and p_l is calculated as follows:

$$p_l = \begin{cases} \dfrac{Q_l^t + T_l - B_l}{Q_l^t + T_l}, \; B_l < Q_l^t + T_l \\ 0, B_l \geq Q_l^t + T_l \end{cases}$$

where B_l is the bandwidth of the link (u,v), T_l is the amount of best-effort traffic observed on the link (u,v) and Q^t_l is the total amount of QoS traffic (i.e., sum of individual QoS level traffics: $Q^t_l = Q^1_l + Q^2_l + ... + Q^n_l$) on the link (u,v). It is crucial that forwarders return accurate (up-to-date) estimates of p_l and d_l to determine precise routes.

- Packet loss measure (p_l) is calculated as previously given where B_l, Q^1_l, ..., Q^n_l and T_l are required parameters for the calculation. OpenFlow protocol enables us to monitor the per-flow traffic amounts (i.e., Q^1_l, ..., Q^n_l where n is the number of QoS levels which can be chosen based on application requirements such as number of different types of services and T_l) on each link. This is done by per-flow counters maintained in the forwarders. The controller can collect the per-flow statistics whenever it requests. The link bandwidth, B_l, is assumed to be known by experimenting or setting manually during the network setup. Note that the packet loss measure does not represent the actual number of packet losses; it is a measure of congestion, which reflects packet loss information.
- Delay is obtained by averaging the observed delay using time stamping (e.g., RTP)

The weight β determines the relative importance of the delay and the packet losses depending on network and traffic characteristics. For large β, route selection would be more sensitive to packet losses on the QoS route. On the other hand, for small β route selection would be more sensitive to delay variation.

The problem can then be formally stated as

$$r^* = \arg_r \min\{f_c(r) | \, r \in R(s,t), \; f_D(r) \leq D_{max}\},$$

that is finding a route r which minimizes the cost function $f_C(r)$ subject to the delay variation $f_D(r)$ to be less than or equal to a specified value D_{max}.

In [5], the aim of this scheme minimizing the maximum link utilization:

$$\max_{l \in L, t = 1,2 \ldots, T} \left(\frac{I_l^t}{X_l}\right),$$

where I_l is the traffic load on link (u,v) and X_l is its link capacity and t represents one of the given traffic matrices. This equation refers to the worst case maximum link utilization for multiple traffic matrices.
For a network with V which is the set of nodes and L that is the set of links, each link $l = (u, v) \epsilon \, L$ has a capacity of R_l which is equivalent to transmission rate of node u. For the case of energy efficient routing in SDN, we propose the following optimization problem:

$$\min C(p) E_i(p) \forall f_i$$

where $E_i(p) = \sum_{s=u}^{d} \sum_{v=s, v \neq u}^{d} E_{i,(u,v)}$ is the number of links for the path p and $C(p)$ is the cost of path p. Each flow f_i has several parameters source s, destination d, start time t_0, and arrival curve.
The constraints are based on to satisfy the throughput and delay requirements for each flow, which can be changed depending on application.

$$D_i(p) \leq d_i$$
$$R_i(p) \geq r_i$$

where $D_i(p)$ is the mean delay of path p and $R_i(p)$ is the throughput of path p. Here $Q_i = [d_i, r_i]$ is the QoS vector for each flow f_i , d_i is the delay requirement and r_i is the throughput requirement.

3.2.1.3 Proposed solution and architecture

We propose an efficient routing and traffic management algorithm to minimize the cost based on power consumption determined as number of active OpenFlow switches in the network under the assumption that the traffic load is low (night time traffic). Power consumption in a network element is equal to sum of power of chassis plus power of linecard plus power of ports. Most power is spent in the chassis. Thus, putting non-used network components to sleep mode reduce network power consumption. We propose an optimum solution using genetic algorithm and a simplified approach while satisfying the throughput requirement of the flows for given capacity of links.

3.2.1.3.1 Network Model

The SDN can be modeled as a graph G < V; Z > where V is the set of nodes and Z is the set of links between the nodes through available ports. $N_{u,t}$ represents the links can be established between node u and node t proportional to the number of ports in the nodes. Each link $(u_y, t_y) \in$ Z has a capacity $C_{(u_y,t_y)}$ where y=1, 2,…, $N_{(u,t)}$. The capacity of each link is adjusted according to the number of available ports in the switches and the maximum supportable capacity of the switch. F is the set of K flows and each flow $f_k \in$ F has several parameters, source, s_{fk}, destination, d_{fk} and throughput requirements, R_{fk} .

The aim of our solution is to find the routes $\mathbf{p} = \{p_{f1}; p_{f2}; \ldots; p_{fK}\}$, where $p_{fi} \in$ Z is the route from source to destination nodes belonging to each flow. The route **p** is found by minimizing the power consumption based on the number of active OpenFlow switches in the network while satisfying throughput requirements of each flow.

Then, the optimization problem can be defined as,

$$\min \; T(\boldsymbol{p}) \qquad (3.2.1.1)$$

subject to

$$C_{(u_y,t_y)\in p_{f_k}} \geq R_{f_k} \; \forall f_k \qquad (3.2.1.2)$$

$$\sum_{y=1}^{N_{u,t}} C_{(u_y,t_y)} \leq C_{u,t} \; \forall u, t \qquad (3.2.1.3)$$

where $T(\boldsymbol{p}) = \sum_{v\in V} x_v(\boldsymbol{p})$ is determined as the total number of active OpenFlow switches in the network after performing routing for all flows.

The first constraint states that the link capacity must be equal to or higher than throughput of each assigned flow. The second constraint denotes that the total link capacity, $C_{u,t}$ between node u and node t must be equal to or higher than demanding rate of total established $N_{u,t}$ links.

Since we assume that the network load is low, the overall network capacity is much higher than the rate requirements of all flows, hence it is possible to shut down a certain number of OpenFlow switches.

3.2.1.3.2 Optimum Solution

In order to find the optimal routing and traffic management, we apply genetic algorithm (GA) which mimics the biological evolution as a search method. GA uses techniques inspired by natural evolution like selection, crossover, and mutation as shown in Figure 2. In order to prevent GAs converging to a local optima, adaptive techniques can be applied by adjusting parameters such as crossover probability, P_c, mutation probability, P_m and population size, M.

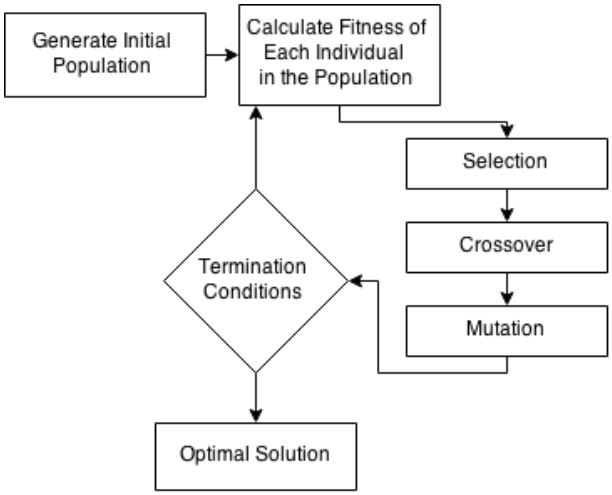


Figure 2 – General Structure for GA

We obtain the optimal routing, $\boldsymbol{p}$ by minimizing T defined in Eq. (3.2.1.1) while taking into account the constraints given in Eq. (3.2.1.2) and Eq. (3.2.1.3) by performing GA.

- Construct initial population: We generate randomly M parents. Each parent includes K different routes, each of which corresponds to a candidate route for the flow. Furthermore, each parent has to satisfy the constraints given in Eq. (3.2.1.2) and Eq. (3.2.1.3).

- Apply selection criterion: We select the best M/2 parents in terms of the number of total active switches as defined in Eq. (3.2.1.1).

- Apply crossover: We apply one point crossover, which is chosen as a midpoint of the corresponding routes of each flow. A single crossover point on two parents is selected and then all values beyond that point are swapped between these two parent values. The resulting ones are added to the population as the new children.

- Apply mutation: We apply uniform mutation method by choosing the value randomly and replacing it with a uniform random value selected in the range of the route-specified upper and lower bounds.
- Convergence: The convergence criterion is selected as unchanged value on the number of total active switches for a certain number of iteration.

3.2.1.3.3 The Proposed Low Complexity Approach

In order to reduce the complexity of the optimal solution, which increases with the number of flows and switches in the network topology, we propose an efficient routing and traffic management algorithm with simplified complexity. In the proposed low complexity approach, the flows are routed sequentially in a random manner instead of routing simultaneously as in the optimal solution. This method also allows for real-time operation where flow requests arrive and are released when the corresponding communication cases.

The optimization problem per each flow $f_k \in F$ can be written as,

$$\min \; T\left(p_{f_i}\right) \tag{3.2.1.4}$$

subject to

$$C_{(u_y,t_y)\in p_{f_k}} \geq R_{f_k} \tag{3.2.1.5}$$

where $T\left(p_{f_k}\right) = \sum_{v\in V} x_v(p_{f_k})$.

After routing the flow $f_y \in F$, the total available link capacity belonging to all links is updated by,

$$C_{(u,t)} = C_{(u,t)\in p_{f_y}} - R_{f_y} \qquad (3.2.1.6)$$

In order to solve this problem given in Eq. (3.2.1.5), we propose a low complexity routing algorithm. Figure 3 depicts the flow chart of the proposed approach.

The proposed low complexity routing algorithm is described in detail as follows::

- *Initialization*: The sets of input OpenFlow switches, I , which are connected to SDN controller and the output OpenFlow switches, O, which are connected to such as base stations for the wireless networks are determined in a given network topology. Then, F-shortest paths for each input and output pair (i, o) are computed and stored in the set $W_{(i,o)}, \forall i \in I \; and \; \forall o \in O$. The set of all routes is initialized by $S = \emptyset$.

- *For the first flow, f_1:*
  - For the flow f_1 with source s_{f_1} and destination d_{f_1}, chose the optimum route p_{f_1} from the set $W_{(s_{f_1},d_{f_1})}$ according to Eq. (3.2.1.4) while satisfying Eq. (3.2.1.5). This optimal route corresponds to shortest path solution for flow f_1 among the possible routes in the set $W_{(s_{f_1},d_{f_1})}$ since it is the first flow to be routed in the network.
  - Update S = p_{f_1} .
  - Update the available capacity of links in the network by using Eq. (3.2.1.6).

- *For k=2, ... ,K*
  - For flow f_k, find the $B \leq F$ routes having shortest paths as $W^b_{s_{f_k},d_{f_k}}$; b = 1,… , B while satisfying Eq.(3.2.1.6)
  - Select the best route for flow f_k in terms of total number of active switches while maximizing the switch usage:

$$b^* = \arg\min_b T^b \quad (3.2.1.7)$$

  where $S^b = S \cup W^b_{f_k}$ and $T^b = \sum_{v\in V} x_v(S^b)$ with b=1,…,B.
  - Update $S = S \cup p^{b^*}_{f_k}$ by adding the new switches for f_k to the set S.
  - Update the available link capacities in the network by Eq. (3.2.1.6).
- *End*

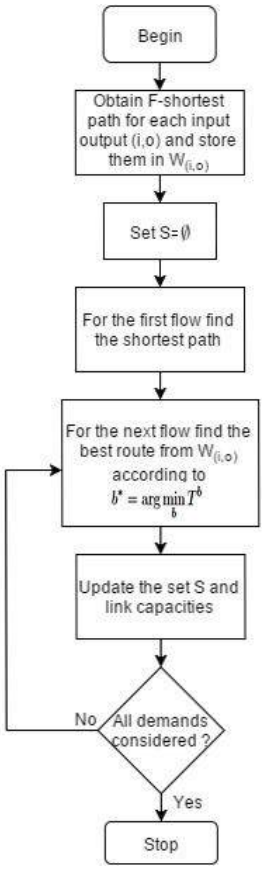


Figure 3 – Flow chart of proposed routing algorithm

3.2.1.4 Main benefits, results & comparisons

We have implemented GA for the topology given in Figure 4. We have assigned total supportable rate per link and satisfied the required rate while reducing the total number of active switches.

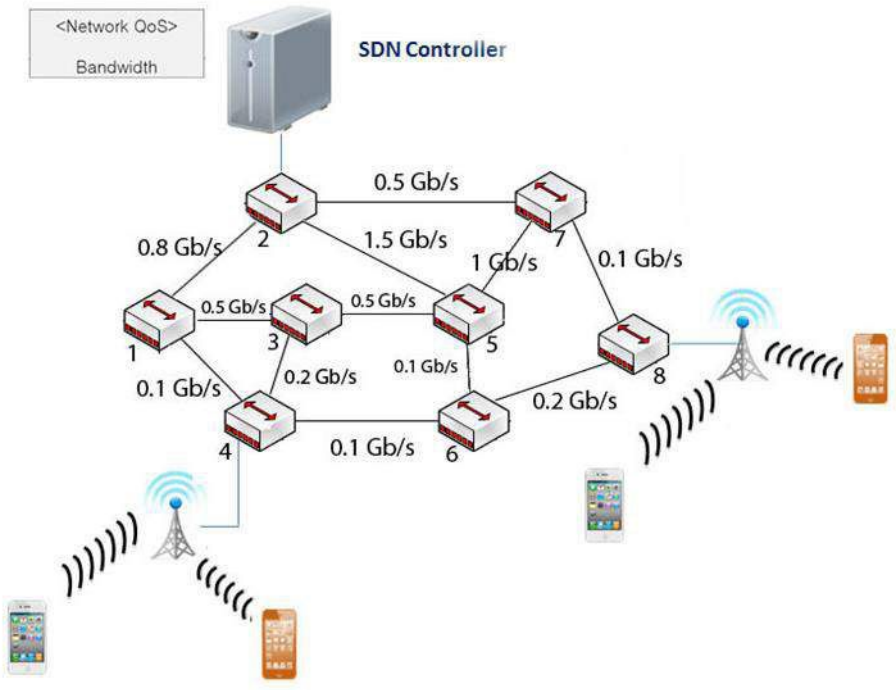


Figure 4 – Example System Topology for GA

Table 1 Genetic Algorithm Outputs

	Source	Destination	QoS Demand	Possible Paths	Chosen Path
Scenario	2	4	100 Mbps	2 1 3 4 2 5 6 4 2 5 7 8 6 4 2 1 3 5 7 8 6 4 2 1 4 2 7 5 3 4	2 1 4

Table 2 Selected Paths for Different Scenarios

	Source	Destination	QoS Demand	Path
Scenario 1	2	4	50 Mbps	2-1-4
Scenario 2	2	4	150 Mbps	2-1-3-4
Scenario 3	2 2	4 8	100 Mbps 75 Mbps	2-1-4 2-7-8
Scenario 4	2 2	8 8	100 Mbps 75 Mbps	2-7-8 2-5-6-8
Scenario 5	2 2	4 4	100 Mbps 75 Mbps	2-1-4 2-1-3-4
Scenario 6	2 2 2 2	4 4 4 4	100 Mbps 100 Mbps 100 Mbps 100 Mbps	2-1-4 2-1-3-4 2-5-3-4 2-1-6-4

3.2.1.5 Recognized issues and future work

As a future work, we will adapt the proposed architecture to software defined mobile networks by sorting the flows in controller. In order to give priorities flows we will consider different parameters such as users' location, traffic model, interference level. Then, we will evaluate the prioritization impact on the performance based on overall power consumption and delay for software defined mobile networks.

3.2.2 Application-Layer Traffic Optimization techniques in SDMNs

3.2.2.1 Definition

Application-layer traffic optimization (ALTO) problem arises when someone is concerned with better-than-random peer selection and/or optimization of rendezvous service for applications fetching distributed content. Typical fields where the ALTO problem occurs are peer-to-peer networks, content distribution networks and datacenters [6].

There is a strong need for cooperation between application and network layers for optimal selection of endpoints. Network operators should be able to provide network maps and cost maps representing distance, performance, and charging related criteria.

CDN and P2P service providers have proper solutions for the ALTO problem. The ALTO protocol [1] has been introduced to provide standard APIs for network, cost map information queries and ALTO guidance service. It enables interoperability between ALTO solutions of different vendors and providers.

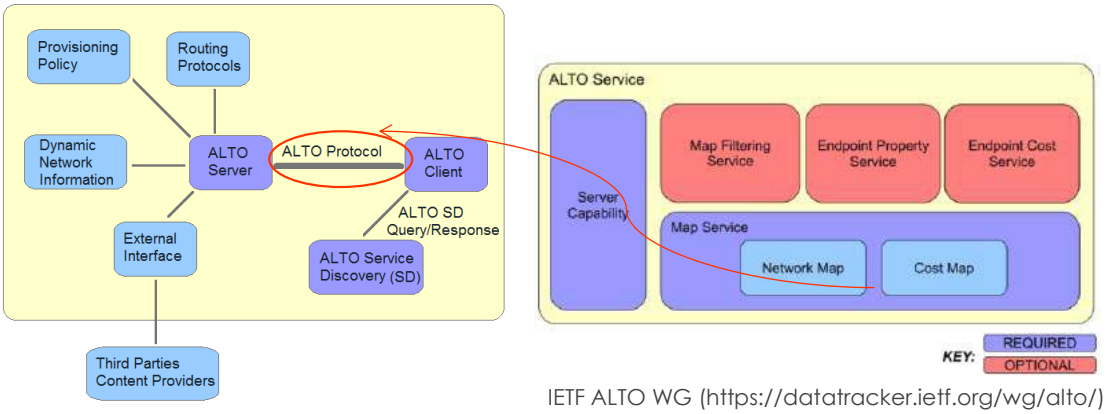


Figure 5 – IETF ALTO protocol.

The role of Application-Layer Traffic Optimization protocol is illustrated in Figure 5. An ALTO client can request network information from ALTO server, related to the ranking of service endpoints under different cost metrics (distance, performance, financial cost etc.).

First Gurbani et al. proposed in [7] the application of ALTO service in the SDN application layer. They argue that the ALTO protocol is a well-defined and mature solution that provides powerful abstraction of network map and network state that can be leveraged by distributed services in SDNs. ALTO hides unnecessary detail of the underlying networks without unnecessarily constraining applications, hence privacy of network information of network operators and content providers can be kept.

To our knowledge, no one has published publications on proof-of-concept implementations of ALTO-SDN integration. Our proposed solution of ALTO-SDN service is described in Section 3. Network information provision from SDN networks to ALTO servers needs to be automatized. ALTO protocol does not take care of this task hence we extend our implementation with this function.

3.2.2.2 Proposed solution and architecture

3.2.2.2.1 Architecture

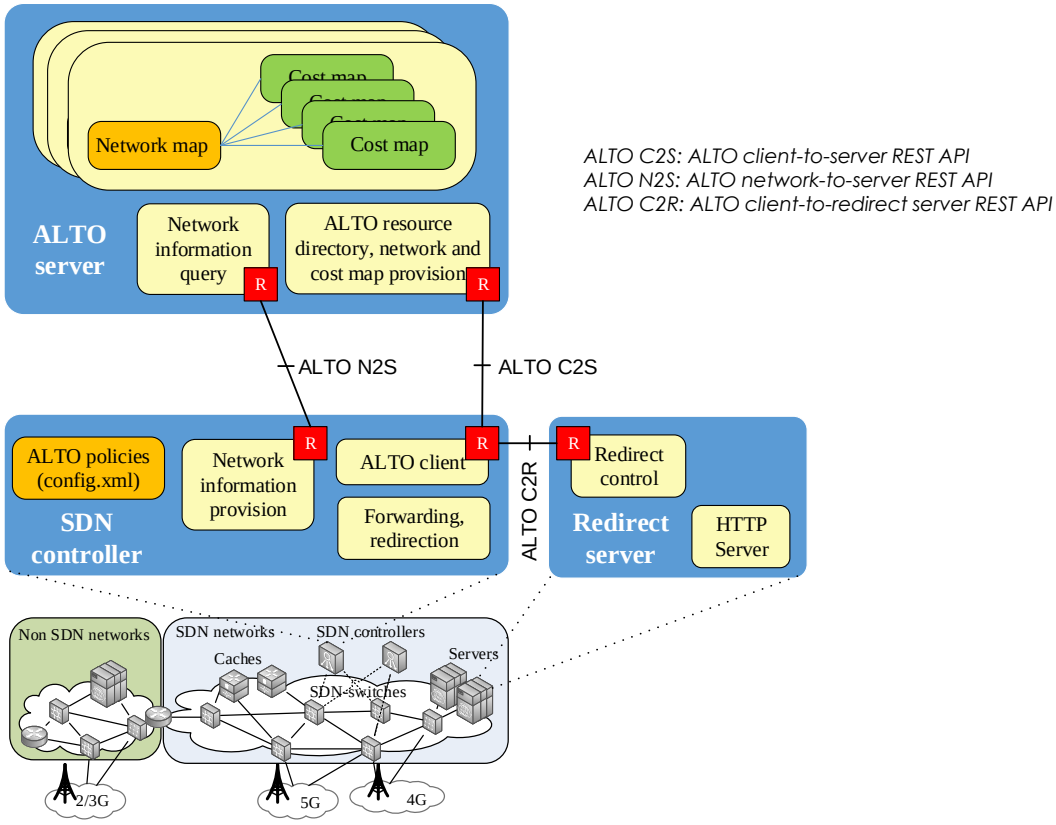


Figure 6 – Building blocks of ALTO-SDN.

Figure 6 depicts the building blocks of ALTO-SDN solution. It includes the ALTO server, ALTO client application of SDN controller, and HTTP redirect server for non-transparent redirection of HTTP requests. Three interfaces are required between these components for proper operation:

- ALTO C2S: ALTO Client-to-Server REST API:

  It realizes part of IETF ALTO protocol messages, including ALTO resource directory, network map, cost map queries and responses and ALTO error message.

- ALTO N2S: ALTO Network-to-Server REST API

  ALTO server requests dynamically network information from the SDN controller through this interface. The SDN controller provides a single-node network view over REST API with JSON-type content

- ALTO C2R: ALTO Client-to-Redirect REST API

  This interface provides the necesseary information to the HTTP redirect server to redirect a specific HTTP GET to the preferred service endpoint selected during the multi-attrbitue decision making process

3.2.2.2.2 ALTO server

The ALTO service must provide at least three basic functions: ALTO information resource directory, network map and cost map provision. The server (depicted in Figure 7) can provide arbitrary number of network maps and related cost maps providing different types of ordinal and numerical cost values assigned to the links between PIDs (endpoint groups). We note that, which network map and which cost types must

be considered for a specific service class is configured by an orchestrator entity (either manually or automatically) using an XML configuration file.

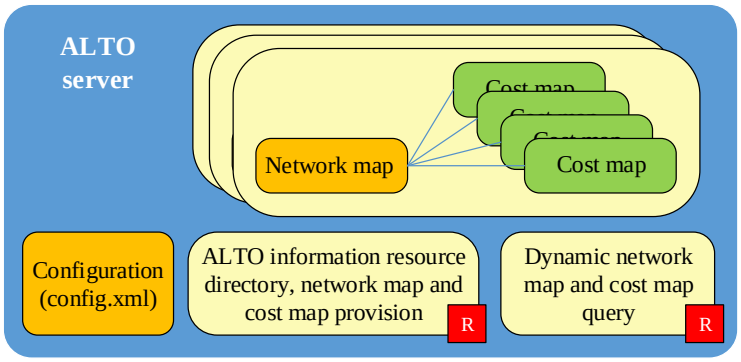


Figure 7 – Main components of ALTO server.

An important change in ALTO-SDN architecture compared to the original SDN architecture is that the selection of the preferred endpoint (decision making) is moved from the ALTO server to the ALTO client. Consequently, in our proposal there is no need for endpoint property querys. In ALTO-SDN, the ALTO server hence mainly is utilized as a pure network and cost map information service. The change in the concept was made due to the fact that it is better to implement communication intensive SDN applications as an application module in the controller.

Another important functionality of the ALTO server is the automatic merging of network and cost map information coming from different sources (over ALTO Network-to-Server APIs), such as CDNs, BGP speakers, SDN controllers. We will focus our research on how SDN controllers can dynamically update the network and cost maps through REST APIs.

In SDN networks, network topology data coming directly from SDN controllers is required. Additionally, traffic engineering data, geo location data, or network resource utilization data could also be used to further refine the maps, or to generate different maps for different service classes. An important requirement is to enable policy driven network information provision, thus the network operator can define the abstraction level of the network map (this requires policy rules both on the SDN controller and the ALTO server side). For Network-to-Server API there exist recommendations for distribution of link-state and TE Information from BGP routers [8][9][10]. A similar approach should be used in case of SDN networks, i.e., the SDN controllers should be able to provide network maps to ALTO servers driven by the SDN network operator policies. The policy rules and APIs should be defined.

Another problem is the calculation and merging of network and cost maps based on the information provided by different SDN controllers, BGP routers, CDN networks etc. The policy rules driving this process and configuring the ALTO client and server side must be carefully designed. [11] defines APIs between CDNs and ALTO servers for dynamic information collection of the states of data centers.

3.2.2.2.3 ALTO client in SDN controller and configuration of operator policies

The ALTO client is implemented as an application module in the SDN controller (illustrated in Figure 8). Its basic functionality is the query of network and cost maps from the ALTO service during the connection establishment phase of distributed services requiring ALTO guidance. It also should store locally in its cache the maps, in order to reduce signaling. Additionally, it also provides ranking of endpoints based on the cost maps obtained from the ALTO server.

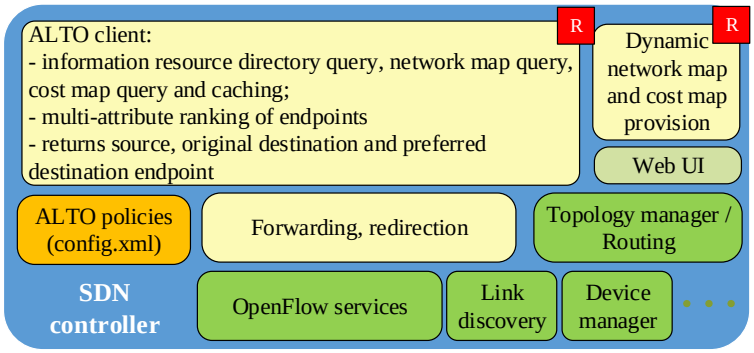


Figure 8 – Main components of SDN controller integrating ALTO client.

The SDN controller must know which service classes require ALTO service. The following config.xml file illustrates the proposed XML schema for the configuration of ALTO policies. The schema includes the definition of service classes, which have a name (id), reachability information of servers (network addresses, port numbers), specification of the related network map and the cost types to be considered for the service. If cost-type is missing then all cost maps should be considered in the ranking of service endpoints. Additionally, the reachability of ALTO server and redirect servers must be given.

In order to reach service endpoints located in external networks, the schema also includes the information on the default gateway, however this information could also be obtained from other controller modules (aware of IP subnets of hosts) or from DHCP services.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<config>
	<services>
		<serviceclass> // multiple ALTO service classes can be defined
		<id>live.mp4</id> // simple id for an ALTO service class
		<server> // available service endpoints (IP and TCP port) for a service class
				<port>6666</port>
				<ip>10.0.0.3</ip>
		</server>
		<server>
				<port>6666</port>
				<ip>10.0.0.6</ip>
		</server>
		<server>
				<port>6666</port>
				<ip>10.0.0.252</ip>
		</server>
		<downlink>true</downlink> // main direction of the service (true: downlink, false: uplink)
		<networkmap>my-default-network-map</networkmap> // network map name for this service class
		<costtype>num-routing;1.0</costtype> // which cost types to consider with how much weight
		<costtype>num-del;0.0</costtype> // cost types: ord-routing, num-routing, ord-del, num-del
		<redirectservers> // ALTO service class will get non-transparent redirection
				<serverRoot>live.mp4/</serverRoot> // The Redirect Server will get redirection request to
				<server> // http://<ip>:<port>/<serverRoot>.
					<port>8080</port> // serverRoot must be the same as the path of the service
					<ip>10.0.0.253</ip> // (e.g., URL path to the video content in an endpoint)
				</server>
		</redirectservers> // if <redirectservers> missing: transparent redirection)
		</serviceclass>
	</services>
	<altoservers> // location of ALTO server
		<serverRoot>ALTOServer</serverRoot> // path of to the ALTO servlet webapp at the Tomcat 6 server
```

```
            <server>
            <port>8080</port> // IP and port of Tomcat 6 server where ALTO servlet is deployed
            <ip>127.0.0.1</ip>
            </server>
        </altoservers>
        <defaultgateway>10.0.0.254/8</defaultgateway> // ';' delimited list of gateways of SDN segments (IP/netmask;IP/netmask )
        <pidmask>255.255.255.255</pidmask> // for the purpose of the demo, endpoints are grouped to provider-defined
</config>   // IDs (PIDs) based on consecutive address spaces using this netmask
```

Two types of redirection operations should be supported by the ALTO-SDN service:

- Transparent redirection: the first type of operation uses flow rewrite rule during the entire period of the service session, causing the client to believe that it communicates with the initially requested service endpoint, while the SDN switch redirects the service data flow(s).
- Non-transparent redirection: this type of operation applies temporary redirection during the connection establishment phase towards an HTTP redirect server until the HTTP redirect server signals to the client the new location of the service. Then the redirected connection is closed by the client and the client reconnects to the preferred endpoint. In this phase the packets of the service data flows are forwarded with the default routing mechanism implemented in the SDN network or virtual network.

Sections 3.2.2.2.4 and 3.2.2.2.5 summarize the operation of transparent and non-transparent redirections. In both cases, the client aims to connect to endpoint A, while the ALTO service prefers endpoint B. In the non-transparent redirection 'R' denotes the redirect server.

3.2.2.2.4 Transparent ALTO-based redirection

Figure 9 presents the sequence chart diagram for transparent ALTO-based redirection of a TCP flow.

The following network entities interact in this flow:

- Client: requests HTTP content
- DNS server: provides IP address resolution based on the fully qualified domain name
- OpenFlow switch: the edge switch of the SDN, interacts with the SDN controller
- Controller and ALTO client: the SDN controller integrates ALTO client module. The ALTO client decides on redirection of flows to the preferred endpoint.
- ALTO server: provides abstract, network and cost map information to the ALTO client using standardized ALTO protocol messages
- Possible service endpoints: A is the endpoint requested by the client, B is the preferred endpoint based on the ALTO guidance.

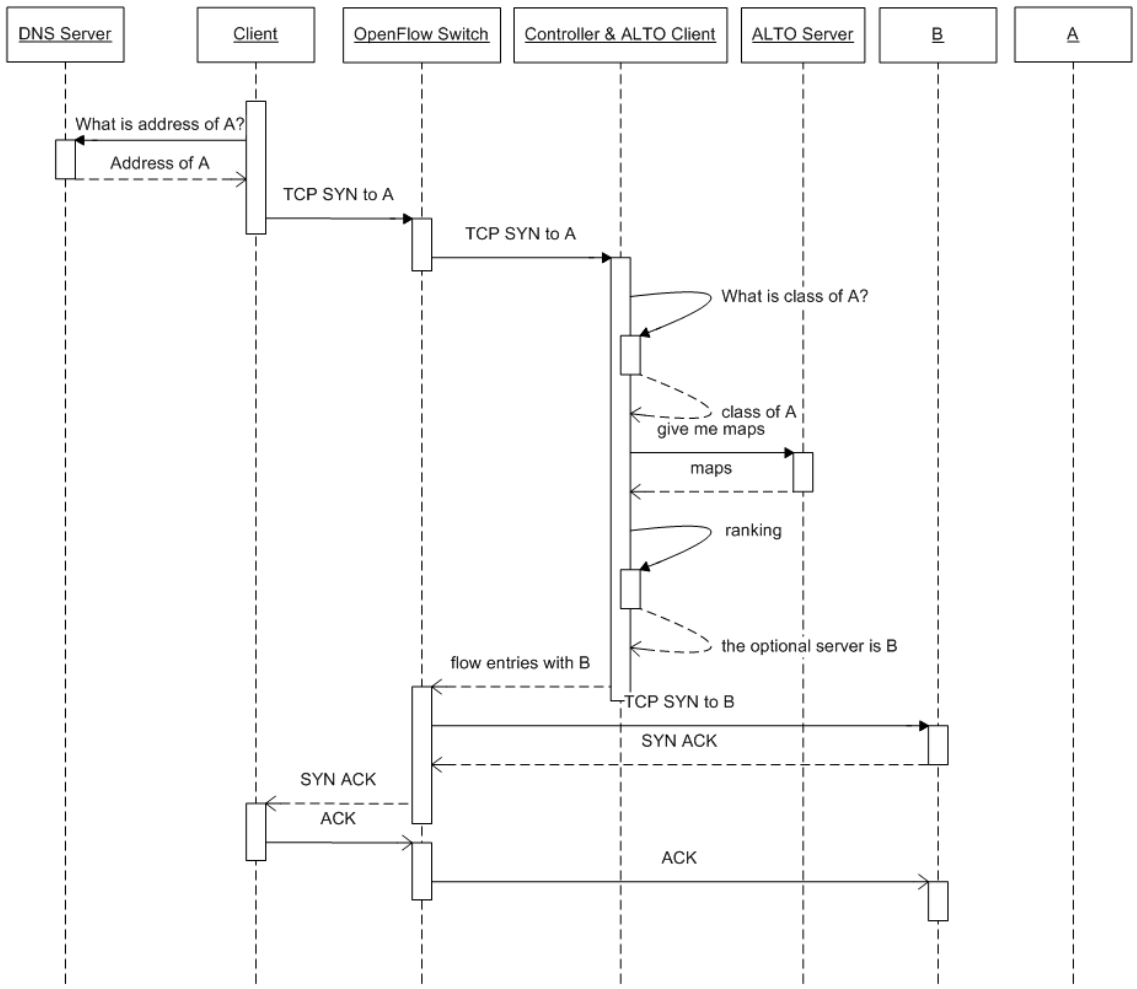


Figure 9 – Transparent ALTO-based redirection.

3.2.2.2.5 Non-transparent ALTO-based redirection using HTTP Redirect server

Figure 10 presents the sequence chart diagram for ALTO-based redirection of HTTP service data flow. The following network entities interact in this flow:

- Client: requests HTTP content
- DNS server: provides IP address resolution based on the fully qualified domain name
- OpenFlow switch: the edge switch of the SDN, interacts with the SDN controller
- Controller and ALTO client: the SDN controller integrates ALTO client module. The ALTO client decides on redirection of flows to the preferred endpoint.
- ALTO server: provides abstract, network and cost map information to the ALTO client using standardized ALTO protocol messages
- Redirect Server (R): the client's request is transparantly redirected to a redirect server, which redirects the client to the optimal content service endpoint.
- Possible service endpoints: A is the endpoint requested by the client, B is the preferred endpoint based on the ALTO guidance.

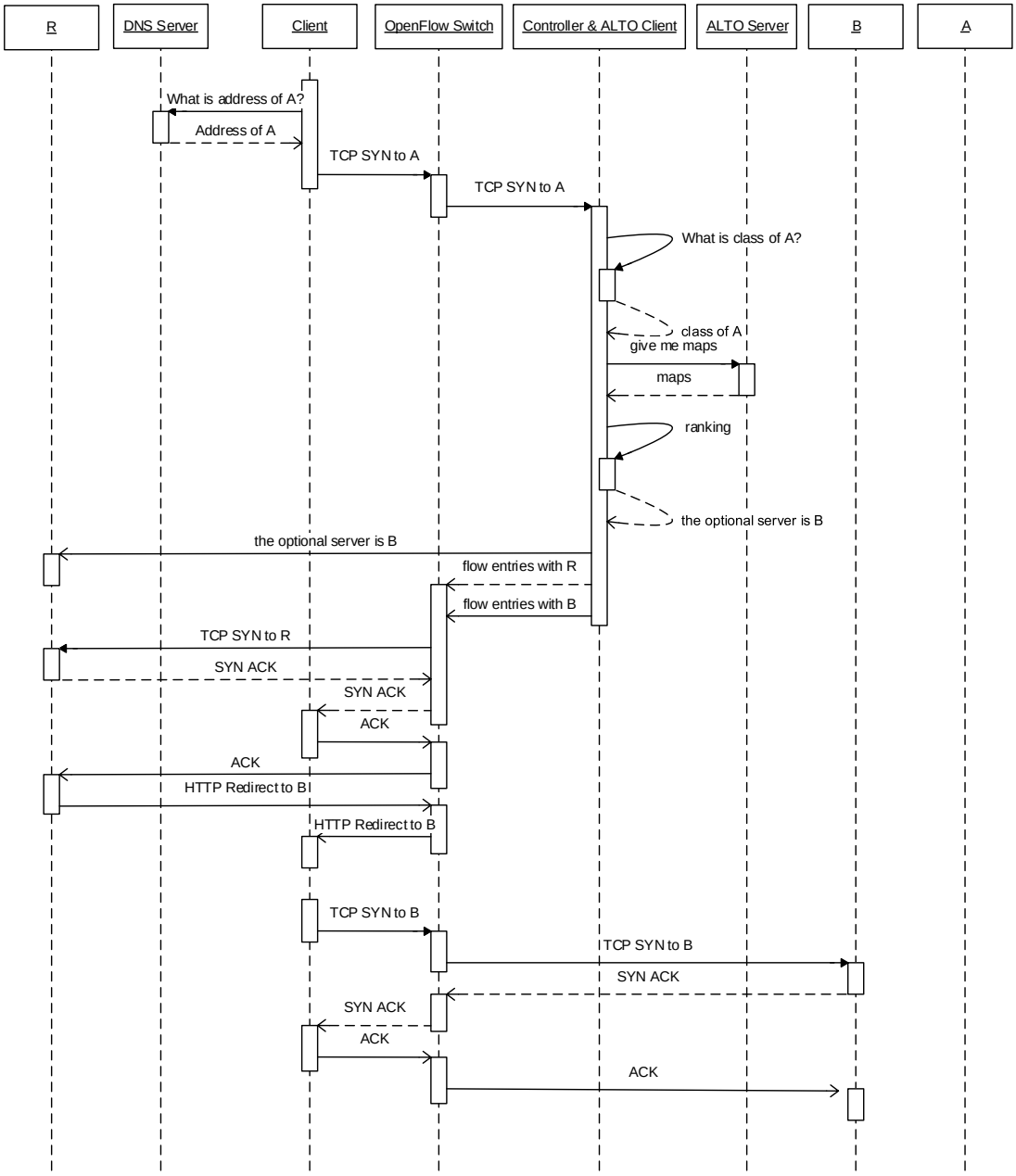


Figure 10 – ALTO-SDN-based redirection of HTTP-based services.

3.2.2.2.6 ALTO Network-to-Server interface: Dynamic network information provision from SDN controller

ALTO Network-to-Server interface provides an essential feature. Through this service, instead of manual configuration of network and cost maps, the ALTO server can dynamically request network information from the SDN controller. The SDN controller provides an up-to-date single-node network view over RESTful interface with JSON media type.

The cost maps are created using the different distance metrics given for each one-way abstract link in the topology structure. Num-routing cost is a metric proportional with the number of switch hops. We also can measure the historical load of the abstracted links by monitoring the increments in switch port statistics (in terms of received, sent or dropped bytes or packets) and deriving distance measures for the abstract link between subnets.

For the derivation of costs between PIDs, we need aggregated distance measures. Clustering techniques apply distance measures between clusters. Such distance metrics can be used in our case as well, e.g., we could measure the minimum, maximum, unweighted or weighted average of the distances between all pairs of hosts in the source and destination subnets (PIDs). We choose to measure the average distance between host pairs. Investigation of all host pairs could be resource demanding, hence a future work could be to select randomly a maximum number of host pairs between two PIDs.

The ALTO server should be able to merge different topologies obtained from different SDN controllers and other sources. Our implementation currently does not support this feature.

3.2.2.2.7 ALTO Client-to-Redirect interface: signaling between the ALTO client and redirect server

Non-transparent redirection applies temporary redirection during the connection establishment phase towards an HTTP redirect server so the HTTP redirect server signals to the client the new location of the service. The redirect server must get the information on the preferred destination from the ALTO client. We defined an HTTP POST message between the ALTO client and the HTTP Redirect to convey the information using JSON format.

After HTTP redirection, the TCP connection is closed by the client and the client reconnects to the preferred endpoint. In this phase the packets of the service data flows are forwarded with the default routing mechanism implemented in the SDN network or virtual network.

3.2.2.2.8 Forwarding function in the datapath layer

SDNs are open for arbitrary routing mechanisms implemented as routing applications on top of existing core services of the controller. For the understanding of provision of "traditional" routing in SDNs, a good overview is provided in [12].

The ALTO-SDN service requires appropriate routing mechanism within and beyond the SDN domain.

The most simple "routing" is when the controller implements the learning switch mechanism. This means that until a switch does not learn the network segment of a destination host, it forwards the frames addressed to this host on all switch ports to all network segments, except the segment where from the frame has arrived (broadcasting). The switch learns the network segment of a host by inspecting the source MAC address in arriving frames, and by binding this MAC address with the incoming switch port in the form of a switching table entry. The disadvantage of this mechanism is that for newly connected hosts, the first exchanged frames towards these hosts are broadcasted in the SDN network. Another disadvantage of learning switch based packet forwarding is the possible presence of cycles in the network graph, causing that packets from a host may come from different ports and also may create broadcast storms.

Therefore, we had to change this mechanism to forwarding mechanism appropriate in SDNs. In this mechanism, once a switch at the edge of the network gets a packet, which does not match with any flow entry, the switch notifies the controller with a PacketIn message. Then the controller calculates the optimal route for the packet and pushes down the appropriate flow entries to the switches on the calculated path. Then, all the remaining packets of the flow are conveyed through this path in the SDN domain.

3.2.2.2.9 Selection of ranking aggregation method

Multi-attribute decision making (MADM) is a sub-discipline of operations research. It provides approaches to handle the complex question of ranking different alternatives under multiple criteria. All MADM techniques require two phases as illustrated in Figure 11. The first phase is the normalization or scaling of performance measures under each criterion (also called, grade assignment, ranking or scoring). The second phase includes the aggregation of the normalized scores resulting in the terminal scores, the ascending or descending order of which reflect the preference order of the alternatives.

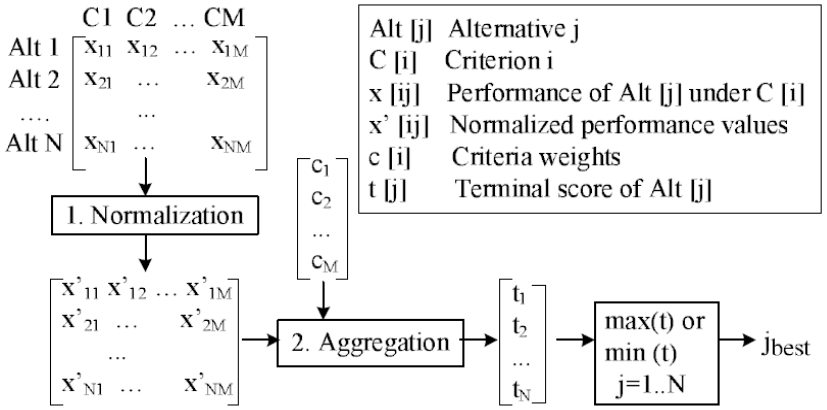


Figure 11 – Multi-attribute decision making.

The normalization phase may be determined by the ranking aggregation phase. In practice, however, the decision makers often alter the normalization phase so as to better describe their preferences. E.g., the Grey Relational Analysis (GRA) [23][24] applies max-min (min-max) normalization by default, but additional normalization techniques have been proposed by Huszák and Imre [24] to mitigate the rank reversal

problem. Certain rank aggregation techniques require only the fulfillment of certain properties by the normalization functions. E.g., the Multiplicative Analytic Hierarchy Process (MAHP) [25] applies geometrical scales, in which the difference between grades, denoted by Δ, means that one of the alternatives is $2^{\gamma\Delta}$ times better in performance than the other alternative under the given criterion. γ, called progression factor of the scale, and the interval of grades can be chosen arbitrarily. In general, someone can define an arbitrary grade assignment function in order to express preferences for performance indicator values.

In certain decision problems, the decision maker may find recommendations that provide information for the specification of normalization or grade assignment functions under certain criteria. E.g., the standard Quality of Service (QoS) characteristics in 3GPP mobile networks [39] make possible the definition of different categories for QoS parameters of service data flows. By appropriate normalization, the obtained grades can reflect in an objective way whether the alternatives are suitable for certain service category or not. These types of normalization are more preferable than the generic normalization techniques. The generic normalization techniques have low chance in suitably describing the preference characteristics of the decision maker.

In point of the ranking aggregation techniques, the Weighted Sum (WS) method [26] is probably the most known and oldest technique. It calculates the terminal score of an alternative as the weighted sum of the normalized scores under the criteria. Weights represent the importance of the criteria. There exist other approaches as well, such as the Weighted Product (WP) method [26], Distance on Ideal Alternatives (DIA) [27], Technique for order preference by similarity to ideal solution (TOPSIS) [28], GRA, Analytic Hierarchy Process (AHP) [29] or the MAHP. The algorithms of these techniques are summarized in Appendix A.1.7.3 of IR3.4b [22].

Multi-criteria decision techniques are incomparable regarding the goodness of the decision because of the lack of knowledge of the ideal decision. Therefore, MADM techniques can be evaluated only by showing certain anomalies in their behavior or by defining properties based on which we can categorize the methods.

A typical anomaly is the rank reversal problem [30]. We are talking about rank reversal when the decision maker takes out or adds new elements in the set of evaluated alternatives, and this fact implies changes in the ranking order of a subset of alternatives, which were elements of the set before. This results in inconsistent ranking order and uncertain decision. Rank reversal may occur due to the fact that the normalization and/or the ranking aggregationfunctions result in inter-dependency between the normalized and/or terminal scores, therefore the ranks of the alternatives. E.g., the functions depend upon the alternatives reaching minimum or maximum performance under the criteria. Alternatively, the removal of an alternative results in the change of ordinal variables under a criterion. As indicated, e.g., by Wang and Luo [30], rank reversal can occur in most of the MADM techniques. The solution for the mitigation of rank reversal anomaly hence is to reduce as much as possible interdependencies between normalized scores by applying a fixed normalization scale or grade assignment function. This is recommended, e.g., by Huszak and Imre for GRA [24]. All of the common normalization techniques summarized in Appendix A.1.7.2 of IR3.4b [22] cause inter-dependencies between the alternatives, which may lead to rank reversal. The MAHP technique defines fixed grade assignment scales, resulting in constant ratio of terminal scores of two alternatives, independently on the actual set of alternatives. The ranking aggregation phase in itself also may cause inter-dependency between terminal scores. E.g., AHP and all AHP-based methods may cause rank reversal [31] regardless of the normalization phase.

MADM methods appear in all disciplines including telecommunication. TalebiFard and Leung [32] apply the WP method for dynamic context-aware access network selection. Song and Jamalipour [33] utilize AHP to derive the weights of attributes based on user's preference and service application, and GRA to take charge of score aggregation for the ranking of the access network alternatives. Bari and Leung [34] proposed a new variant of TOPSIS for access network selection in heterogeneous networks, which solves the rank reversal problem found in TOPSIS. Hager [35] and Johnsson [36] use a modified AHP for algorithm selection for user authentication. A more recent application area of MADM is the design of context-aware media independent handover services [37][38]. Ghahfarokhi and Movahhedinia [37] mainly focus on the collection of contextual information for the decision, but also propose a handover decision framework, which could include any MADM in the ranking aggregation phase. Their proposal includes WS, WP, GRA and TOPSIS, but does not provide guidance on the selection of decision engine. Ranking is also used for peer/server or cache selection in case of distributed services.

ALTO requires the application of MADM techniques for the ranking the endpoints based on multiple cost types, such as routing costs, link utilization costs, available capacity, and round-trip times for E-E paths.

It is clear that currrent MADM applications in the field of telecommunication do not provide guidance in decision engine selection. Hence, our objective is to analyze the impact of ranking aggregation tecniques on the decisions and provide guidance in the selection.

First, we will show that ranking aggregation technique selection may have influence on the final decisions. The demonstration is described in the validation results in Section 3.3.5.6.1 of IR3.4b [22].

Then we show a methodology to form clusters of different techniques based on the similarity of the decisions.

3.2.2.2.9.1 Clustering ranking aggregation methods based on the similarities of their ranking

Next, we present a new method for clustering ranking aggregation techniques based on the similarities of their rankings. Using the method, we can determine interchangeable ranking aggregation techniques, and we can provide guidance on decision engine selection in case of MADM problems.

The method and the results of its application are described in Sections 3.3.5.6.2 and 3.3.5.6.3 of IR3.4b [22]. The main steps of the method are illustrated in Figure 12.

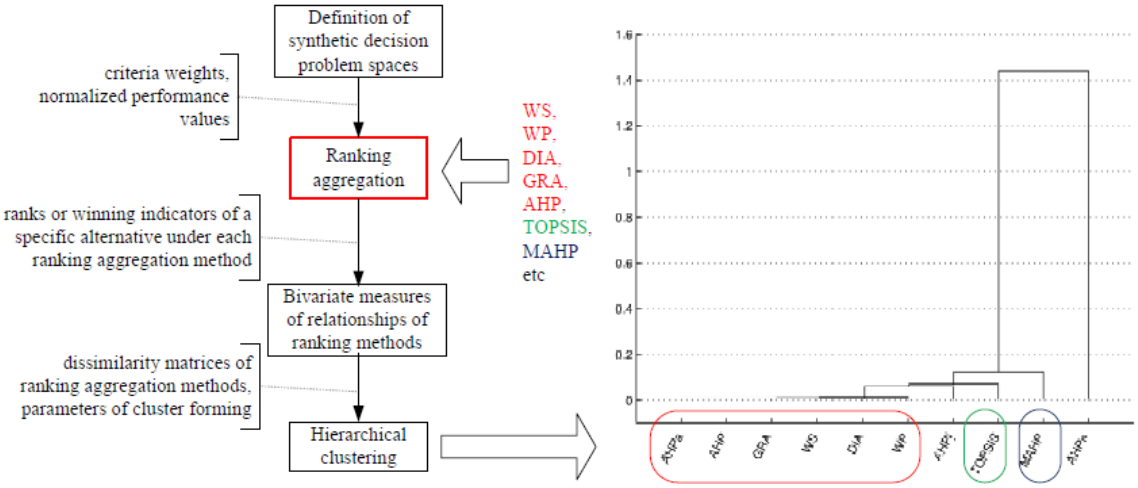


Figure 12 – Main steps of clustering ranking aggregation techniques.

Dissimilarities between the ranking aggregation techniques are not easy to express, because there are too many problem types to deal with, depending on the number of alternatives, number of criteria, criteria weight assignment, set of decision cases, normalization function. Therefore, we applied statistical approach for the determination of dissimilarities. We generated synthetic problem spaces with different number of alternatives and criteria, using different criteria weight vectors for the comparison of the winning indicators and ranking of alternatives in case of different ranking aggregation methods. We calculated measures of bivariate association between the ranking techniques, and based on those measures, we formed clusters of similar methods using hierarchical clustering.

3.2.2.2.9.2 Discussion of results

Our results show that the selection of ranking aggregation technique has influence on the final ranking of alternatives. However, the dissimilarities between the ranking aggregation techniques are not easy to express, because of the wide set of possible characteristics of decision problems. Therefore, we created scenarios with synthetic decision problem spaces and applied simulation-based evaluation instead of analytical approach for the comparison of ranking aggregation techniques. Due to that, the obtained results are valid under the following condition. In our scenarios the normalized performance values fell roughly in the interval [0,1]. We remark that the dissimilarities between the methods may differ if the normalization resulted in values of different intervals. However, most of the compared techniques are typically used with such normalization function, which results in normalized performance values that fall roughly in the investigated interval.

We applied sum normalization before the ranking aggregation phase in the calculation of bivariate association measures and during hierarchical clustering. By fixing the normalization technique all possible normalized performance matrices could be easily covered with a grid of matrices. Therefore, all types of concordant, discordant, tied relationships between pairs of observations of ranking could be revealed under that grid. In the third scenario, this approach became inefficient due to the large number of possible decision cases, therefore random decision cases were generated, which cover only by chance the set of possible discordant and tied relationships between pairs of observations.

The dissimilarities between the methods depend also upon the number of alternatives and criteria. It was evident in every scenario that Hager's AHP method (AHPh) operates in a highly inconsistent way related to the other methods (i.e. its cophenetic distance from the other methods fell roughly in the interval [1.1,2]). The illustrative examples on the odds of winning, the negative correlation and association measures indicate

that AHPh should not be used because it typically assigns higher ranks for alternatives showing lower performances.

The dissimilarities between the rest of methods were revealed only in case of the scenarios where there were more than two alternatives and criteria. The results showed that MAHP method was the second most dissimilar from the rest of the methods (i.e., its cophenetic distance was roughly in the interval [0.1, 0.8] from the rest of the methods). Its dissimilarity is due to the way of operation of MAHP. MAHP first calculates the differences between the normalized performance values, but then treats them as logarithmic differences. The grades given to MAHP should indicate logarithmic steps in terms of performance. The philosophy behind MAHP is that human decision makers notice differences between alternatives, typically, when the performance ratios are enough high, i.e, 2^{γ}. That is reflected, e.g., by the dB metric [25]. Therefore, the dissimilarity of MAHP is not a conceptual fault. It basically reveals another important property of MADM methods, i.e., the interpretation of normalized scores. We remark that it is hard to decide which is the better approach in the matter of normalization or grade assignment, however, it is evident that those normalization techniques are more preferable, which describe more accurately the preference characteristics of the decision maker. Therefore, the design of normalization phase must get enough attention in any application of MADM techniques.

The third most dissimilar method from the rest of the techniques was TOPSIS. Its cophenetic distance from the rest of the methods was around [0.05, 0.2] considering Kendall's τ_c and Spearman's ρ. Its dissimilarity may be due to the way of calculation of terminal scores, given in Equation (A.1.7.21) in Appendix A.1.7.3 of IR3.4b [22]. AHPj was the fourth most dissimilar method in relation to the rest of the methods. Its cophenetic distance from the rest of the alternatives was approximately [0.01, 0.06]. Its dissimilarity may be caused by the definition of pairwise performance ratio matrices given in Equation (A.1.7.27) in Appendix A.1.7.3 of IR3.4b [22]. If $X'_{j,i} \geq X'_{k,i}$ then it calculates with the absolute difference of the performances of the compared alternatives, on the other hand, when $X'_{j,i} < X'_{k,i}$ then it defines a hyperbolic function of absolute differences, which results in an asymmetric characteristic of the preference ratio function. The rest of the methods, i.e., WP, WP, DIA, AHP, AHPa were in all scenarios mapped into the same cluster. Their cophenetic distances fell roughly in the interval [0,0.01].

Consequently, based on the previous results, in scenarios, which have roughly ten criteria and ten alternatives, the following methods are interchangeable: WP, WP, DIA, GRA, AHP and AHPa. Therefore the less computationally demanding method is recommended for use, i.e., WS. Furthermore, MAHP follows a different concept, but it can also be selected for use. In case of MAHP the decision maker must be aware of the fact that the grades indicate the ticks of a logarithmic scale. TOPSIS and AHPj are less preferable because their dissimilarities from the previous methods are probably due to conceptual issues. Finally, AHPh should not be used, due to its evident ranking errors.

3.2.2.2.9.3 Conclusions for ranking aggregation method selection

By applying our method to ten different ranking aggregation techniques, we showed that with respect to the ordering of alternatives, practically, there is no difference between many of well-known techniques in small decision scenarios having roughly ten alternatives and ten criteria, which is an average size of MADM problems in the field of telecommunication. This statement was not mathematically proven but shown by simulated decisions.

An important assumption was that the normalized performance values fell in the interval [0,1], i.e., the decision makers applied sum or max normalization before ranking aggregation. The results would not hold in case of other intervals; however our clustering method could show also in such cases, which ranking aggregation methods are interchangeable due to high similarities.

Under our assumptions, the simple WS method can substitute most of the methods, including AHP, DIA, GRA, WP methods. Using different dissimilarity measures, we showed that some ranking aggregation techniques are dissimilar from others. Such methods are the MAHP, Hager's AHP variant, TOPSIS and Johnson's AHP variant.

Therefore, we recommend to apply the Weighted Sum method in the ranking aggregation phase of service endpoint selection in ALTO-SDN use case.

3.2.2.3 Main benefits, results & comparisons

ALTO service integration into SDNs can result in several benefits for mobile network operators for the orchestration of endpoint selection for distributed services. ALTO service provides appropriate level of abstraction of network and cost maps, enforcing the policies of mobile network operator and optionally other actors, but keeping the privacy of network topology information. SDN controllers can enforce flow redirection and can dynamically provide abstracted network and cost maps to ALTO server.

ALTO-SDN service could be an enabler for cross-stratum and cross-domain orchestration of endpoint selection for distributed services in SDMNs. By manipulating cost maps, it could become an efficient tool to enforce the policies of mobile network operator and maybe other actors in different use cases, such as in-network cache, CDN server, breakout point selection or virtual network function instance selection.

The main benefits of moving ALTO client to the SDN controller are hence the following:

- ALTO guidance is decoupled from the application. The operator policy determines dynamically, which services should get endpoint selection guidance.
- Reduced deployment cost of ALTO service. No additional deployment costs of ALTO clients in user equipments, trackers, resource directories.
- SDN paradigm enables dynamic, abstracted network information provision from SDN controller to ALTO server about the controlled SDN network clusters.

ALTO-SDN solution can serve other traffic management solutions in terms of service endpoint selection. The main interaction point with ALTO-SDN service is the policy descriptor file, which can be updated either automatically or manually by the Operations Support System (OSS) or ETSI NFV orchestration processes. Possible interworking scenarios with other traffic management solutions in the SIGMONA project are the following.

- Combined optimization for virtual mobile network service chain network embedding and topology planning might be a top-level orchestration function. This optimization process can be triggered by certain types of events by monitoring applications or it can be scheduled periodically. It results in the decision on the placement of available service endpoints, i.e., different types of VNFs realizing the 3GPP mobile network core control plane or the some value-added services of the operator. In case of distributed functions, ALTO-SDN can be an enforcer to direct the clients to the appropriate VNF. Hence, after the combined optimization gives us the results on the location of VNFs and the virtual network topology, orchestration could also update ALTO-SDN policies, i.e., the list of available endpoints. ALTO network and cost maps will be updated automatically, if SDN controller sees changes in the topology of virtual networks.
- Optimal video delivery: ALTO-SDN could be used for the selection of proxy or video delivery server, which are distributed.
- Joint traffic and cloud resource management for service chaining: ALTO-SDN culd be used for the enforcement of the selection of optimal VNFs in case of distributed services
- ALTO-assisted flow management (any mobility mobility management scenario): ALTO-SDN could be used during handover decision / interface (RAN) selection, running as an application of the SDN controller. The handover decision engine could also take into consideration ALTO network and cost maps, during the selection of next access network and serving gateway of the user equipments.

The description of our proof-of-concept implementation has been published in [13].

3.2.2.4 Recognized issues and future work

Our ALTO-SDN implementation applies reactive flow management. Originally, our main focus was on the northbound APIs of the controller, and we just applied the basic flow management possibilities provided by the chosen SDN controller (Floodlight). Therefore, we applied reactive flow management using OpenFLow 1.0, which can only inspect TCP flows, but not specific TCP control messages. ALTO-SDN could be re-designed to work in proactive way, using OpenFlow 1.3, which can already match on TCP SYN packets. Section 3.2.2.4.1 describes the draft of a new table pipeline structure, which segregates L3 routing, Diffserv QoS enforcement and ALTO redirection functions, and enables proactive setting of flows by different SDN applications on top of the SDN controller.

ALTO server should be extended to get and merge network information from multiple network controllers, furthermore, ALTO server requires other APIs to get cost maps from service/media content providers, such as content distribution networks, in order to consider the preferences of all actors during endpoint selection. Multi-attribute decision-making is just one possible way to rank the alternatives, policy rule-based decision engine could also be appropriate for tanking the endpoints.

We noticed that abstract network and cost maps are good for keeping the privacy of network topology, but in some cases abstraction influences badly the redirection decisions. This is due to the fact that costs reflect distances between node clusters, not between specific endpoints. The granularity of redirection decisions is constrained by the size of the provider-identified domains.

Regarding ALTO assisted mobility (flow) management as a future use case; currently ALTO network domains are differentiated based on IP prefixes. We could extend the defintion of provider-identified

domains using other parameters, e.g., border switch location, Routing Area Identifier, SGSN/S-GW identifiers, eNodeB identifiers etc, however this requires that information on these data are available for the flows in SDN switches, or the SDN controller should be able to get these information for different flows.

3.2.2.4.1 Proposed OF table pipeline structure for proactive flow management

As a future enhancement of ALTO-SDN solution, the ALTO client application could request ALTO network and cost maps periodically or due to certain trigger events for each ALTO service class. Hence, it could install proactively the necessary flow entries in the SDN switches. This would replace the reactive flow management solutions proposed in Sections 3.2.2.2.4 and 3.2.2.2.5.

Another enhancement proposed in this section is the integration of Routing and DiffServ QoS control and ALTO-SDN operation at the level of table pipeline processes.

Next, we assume a network scenario, where the ALTO redirection is enforced in the first IP GW of the UEs. This GW is also in charge of L3 routing and DiffServ QoS services. Figure 13 illustrates a possible table pipeline structure in the first IP GW. This structure could result in fast packet forwarding and scalable operation in case of millions of flows, due to the functional splitting of the tables, and reduction of flow entries. If flows were proactively set by the related SDN enabler applications, this table pipeline structure could also result in reduced PacketIn rate towards the SDN controller.

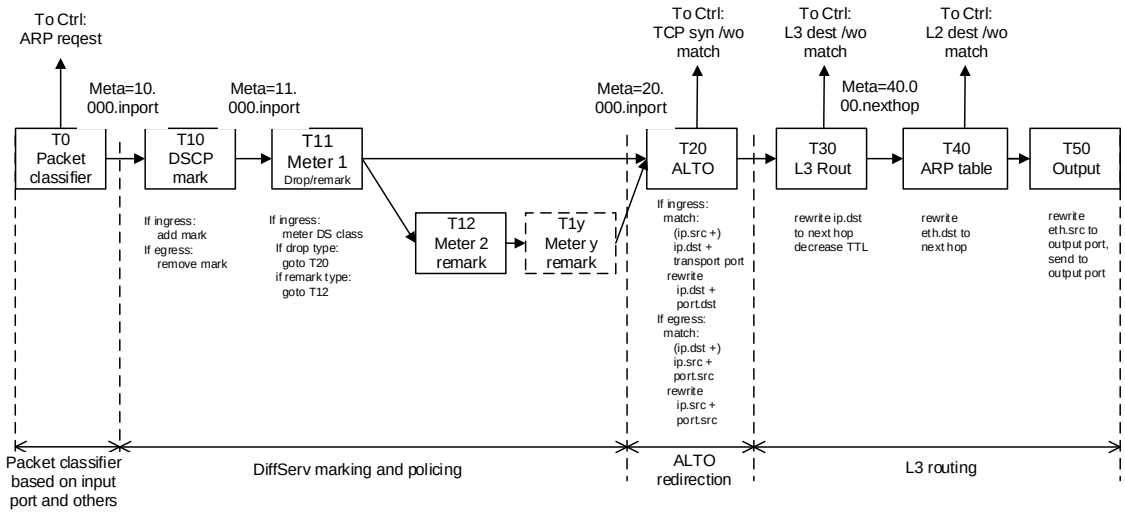


Figure 13 – Possible table pipeline structure for the realization of DiffServ QoS, ALTO redirection and L3 routing in the first IP GW of the UEs.

The table pipeline process includes the following tables:

- Table 0 classifies packets based on input port (and possibly other parameters) in order to decide if it has arrived from or is destined to a UE / or in general, the service claimant entity. It matches on in_port and L2, and sends the packet either to the controller, if it is an ARP request, or to the next table. The controller may create an ARP reply if it is the recipient of ARP request, and send back the ARP reply on the input port. The in_port is written in the Metadata, in order to signal if it is ingress traffic, coming from the UE and needing e.g., DSCP marking, or egress traffic, going back to the UE and needing e.g., removal of DSCP mark.
- Tables 1x are in charge of DiffServ marking and policing. Table 10 adds/removes DSCP marks, by matching on Metadata (ingress/egress) and n-tuples. Table 11 sends the flows to rate meters, if the packet matches on Metadata (ingress) and a specific DSCP mark (and possibly n-tuple). If the meter is of drop type, the packet is sent to Table 20. If the meter is of type DSCP remark, the packet is sent to the next table, Table 1y, y=2..9. Similarly, to Table 11, this table sends the packet to a rate meter, if it matches on the previously remarked DSCP value, Metadata (ingress) (and n-tuple). Egress traffic, going out of the DiffServ domain, should not be processed by Tables 11-19, just by Table 10. We note that multiple branching for DSCP remark type meters and flows are possible, e.g., one branch could be used for each Assured Forwarding class (i.e,, AF1y, AF2y, AF3x, AF4y) containing three tables for each drop precedence. The necesseary flow entries could be managed by the DiffServ QoS application of the SDN controller.
- Table 20 is in charge of ALTO-based redirection of flows. Given the location of first GW in the topology, the actual ALTO policies (i.e., filters for destination IP address, next header, destination port and possibly source IP address, virtual network identifiers etc), and the current ranking of

available service endpoints under each ALTO service class, the ALTO client application of the SDN controller could install proactively the necesseary flow entries in the first IP GWs of the UEs. TCP Syn packets not matching the proactively set flows should be either dropped or sent to a default service endpoint or sent up to the controller for further decision.

- Tables 30, 40 and 50 realize the necesseary packet header processing for L3 routing. Table 30 realizes L3 routing table: it contains flows, which match on virtual network ID, destination IP address, Metadata, and is in charge of setting the next hop in the destination IP address and decreate TTL. Table 40 implements ARP table, and is responsible for setting the destination MAC address of the next hop. Table 50 is in charge of sending the packet to the right output port and for setting the source MAC address to the address of the output port. In case of a packet does not match the flow entries, the packet is sent up to the controller for further processing, e.g., for reactive flow entry writing in the switches. The routing application of SDN controller is responsible for the managment of Tables 30, 40 and 50.

In our future work, this table pipeline procedure will require further tuning, and harmonization with other traffic and mobility management solutions. It seems that it could result in reduced PacketIn rate compared to reactive flow management, because only those TCP syn packets must be sent up to the controller, which do not have the necessary flow entries.

3.2.3 Joint traffic and cloud resource management for service chaining in SDMNs

3.2.3.1 Definition

An explosive growth in the data traffic necessitates higher capacities in the mobile network infrastructure. Thus, network is becoming more difficult to manage. In this situation, operators try to meet users' and enterprises' expectations with higher performance, ubiquitous connectivity, lower operational cost. Services provided by operators have to become more efficient to handle competitive market and fast rollouts are required.

Data services require many network applications such as firewalls, content filters, intrusion detection systems (IDS), deep packet inspection (DPI), network address translation (NAT), content caches, load balancers, wide-area network (WAN) accelerators, multimedia transcoders, logging/metering/charging/advanced charging applications, etc. which already exist in today's networks. Such applications are generally referred to as middleboxes, because they are commonly executed along the traffic path, with their existence usually unknown by the end users. In fact, almost all traffic in mobile networks visit a pre-defined sequence of middleboxes en route to its destination today. Such a sequence is commonly referred to as a "service chain." Then, a service chain consists of a set of network services that are interconnected through the network to support applications such as VoIP, streaming video, e-mail, web browsing, etc. In other words, service chaining "orchestrates" the network flow for every offered application.

There are still no protocols or tools available for operators to perform flexible, dynamic traffic steering. Solutions currently available are either static or their flexibility is significantly limited by scalability inefficiencies. Creation of high performance service-chaining applications is essential to meet traffic demand, higher quality expectations from customers while reducing capital and operational expenses associated with their networks. To achieve this, only necessary middleboxes should be processed, should refrain processing unnecessary middleboxes. Detailed identification of each flow, and establishing an appropriate dynamic service-chain for that flow is the main problem.

Greater control over traffic and the use of subscriber and flow-based selection of virtualized network functions and inline services can lead to the creation of new offerings and new ways to monetize networks. Dynamic service (both virtualized network functions and inline services) steering enables operators to realize scalable network function components in the cloud while providing the capability to offer subscribers access to products such as virus scanning, firewalls and content filters through an automatic selection and registration portal.

There is no existing body of work that jointly and dynamically optimizes the cloud realization of functions and services and a service-chaining route for a given network flow. Related work until now concentrates on efforts of either cloud management or service chaining for SDN networks. The important part of this project is cloud and virtualization environment with SDN concepts. Ericsson Turkey will focus integrating SDN controller and cloud manager to optimize service-chaining application with NFV. For instance,

middleboxes may be in servers with SDN domain or they may be virtualized network functions in cloud environment. In SDN domain, we try to communicate with Network management system and try to do service chaining in this domain. In cloud environment, we try to communicate with cloud manager and try to do service chaining in this domain with cross-domain orchestration concepts. Addition to general situation, service chaining forwarding path will be calculated according to routing path load, service load and determinated QoS parameters.

3.2.3.2 Proposed solution and architecture

Ericsson Turkey's solution about joint traffic and resource management for service chaining in SDMN, enables management of flows in SDN network through a centralized controller. Thus, behavior of forwarding plane can be modified more dynamically and can be configured based on congestion information about network in terms of traffic management. The approach of choosing a set of service functions and creating an individual path for each data flow is called service chaining.

A service chain consists of an ordered set of service functions. Ericsson Turkey Service Chaining solution provides the following:

- Configure Service Chain
- Configure service chains according to network statistic information
- Configure service chain according to traffic types
- Program the forwarding planes according to the service chain configuration

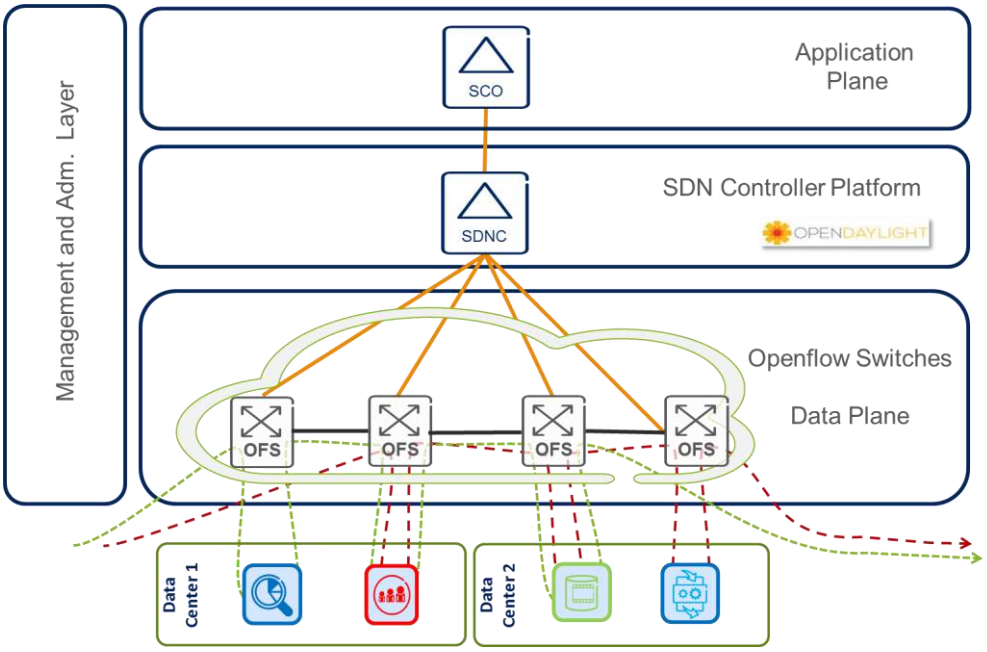


Figure 14 – Service Chaining Orchestrator (SCO)

Ericsson Turkey SDN Service Chaining solution complies with the ONF Software Defined Network Architecture. The architecture consists of distinct layers (data plane, controller platform, application plane and management and administration) as can be seen in Figure 14.

Service Chaining Orchestrator (SCO)

In Ericsson Turkey solution Service Chaining Orchestrator (SCO) is one application, which provides "Services Function Chaining" (SFC) in the Application Plane. The Service chaining app that stores the forwarding information in a high level detail and configure chain, and sends this information to the controller platform and also provides following functionality in addition to listed above:

- May communicate with other applications such as DPI, PCRF, etc.

- It generates service chain by communicating with management layer (OSS/BSS, Cloud Man.) and therefore create additional efficiency

Controller Platform

The Control Plane contains the SDN Controller. In Ericsson Turkey, solution the SDN Controller is OpenDaylight. The SDNC is dynamically aware of the network topology. Controller platform gets the input (service chains input) from "Our Module" and translates this input into a language understood by OpenFlow network elements.

- It manages the transport connectivity provided by the Data Plane
- It offers an abstract network view to the Application Plane
- It receives network control requests from the Application Plane and
- translates it into forwarding instructions understood by the Data Plane

Data Plane

The Data Plane consists of SDN forwarding elements (devices). These nodes perform packet forwarding according to rules received from the SDNC. Ericsson Turkey has selected the OpenFlow protocol to manage the SDN forwarding plane. The OpenFlow switch (OFS) is the forwarding element that is able to apply the actions, which come from SDNC.

Man. And Adm. Layer

The Management & Administration Plane supports interfaces for node and network management. To configure service chain path, SCO need to communicate with this layer. Management & Administration Plane can be a Network Management System (NMS) or a Cloud Orchestration Layer.

Topology and interfaces

Figure 15 shows the network topology and the integration of the SDN Service Chaining solution SCO into the network. SCO, SDNC and OFS are the central network elements of the Ericsson Turkey SDN Service Chaining solution.

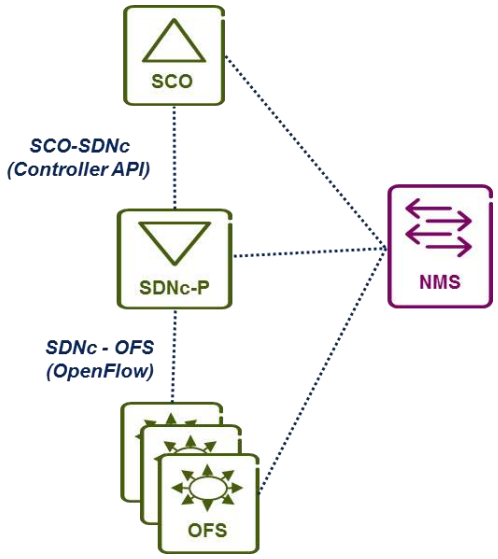


Figure 15 – SDNC

NMS provides performance information, inventory information in our solution.

For performance information, NMS and SCO collects load statistic information periodically from network. Then SCO can use this information to configure service chain paths. Additionally, NMS can generate

"alarm" for congestion situation or exceed threshold situation and SCO can manage the chain paths based on this information considering the traffic types.

NMS also provides the inventory information, topology information about SDN, non-SDN network and other service function nodes. This information is used by SCO to create service chain sets.

The interface between SCO and SDNC uses controllers API (interfaces) and northbound API. Northbound API might be REST stands for Representational State Transfer. It uses the four HTTP methods Get, Post, Put and Delete to execute different operations.

SDNC and OFS interface is OpenFlow. OpenFlow is an enabler for SDN. It is an open standard protocol between the control and forwarding planes. It is one of the most-widely used protocols in SDN, but it is not the only alternative.

The main task of SCO is creating chain paths and setting flow rules to SDNC via controllers API (interfaces) and northbound API. SDNC translates these to Match and Action rules which are entered to the forwarding elements. By doing this, forwarding elements forward the packets according to the match-action lists. By using the information, which is provided by NMS, SCO can configure and reconfigure flow rules.

3.2.4 SDN-controlled device-to-device communications

3.2.4.1 Definition

Wireless operators have to extend the capacity of their 4G cellular network to cope with the growing mobile traffic. Therefore, the challenge for the operators is to find ways to insure a good tradeoff between the network performances and its infrastructure financial cost. The answer may be getting wireless traffic off expensive wireless networks quickly using 4G network offload. Offloading 4G radio network could be insured through deploying Wifi hotspots and through the use of the femtocells. These solutions are nowadays commercialized by the operators and could create under a big shared cell, many private areas with higher performances and more privacy for exchanging data between users in the same LAN.

The femtocells are small local cells with a limited area that share the same frequency resources than the macro cell and are managed similarly to the traditional 4G cells. The users under the coverage of the femtocell are connected to it and their traffic could be redirected to the macrocell infrastructure through a backhaul link between the femto HeNB and the macro eNB. The Wifi hotspots deployment, seems to be the favored strategy as it may not increase at all infrastructure costs and could connect more devices as the most of the nowadays commercialized devices (laptops, tablets, netbooks, etc.) come with Wifi interface.

Many operators already provide WiFi services commercially in such hotspot zones, so using WiFi as a 4G network offload strategy may not increase infrastructure costs at all. Deploying hotspots can allow wifi devices to use the 4G services indirectly. This method has also additional benefit for both the operators and the mobile users:

- Extending the cell coverage
- Avoid congestion in the Ran infrastructure
- Aggregation of an additional bandwidth, which increase the average cell capacity.
- Growing number of connected devices
- Providing private infrastructure for entities with more security than the public network.

Offloading the 4G traffic could be a good solution to alleviate the traffic density at the RAN level but it could also increase the complexity at the backhaul level as the network has to ensure a management of the resources of different technologies simultaneously which are collocated at the same area.

3.2.4.2 Proposed solution and architecture

Adressing the objectives described in Section 2.2.4, this use case investigates opportunities to offload 3GPP Radio Access Network by the use of IP Wireless Mesh Network using SDN capable smartphones. CEA LIST is developing a new SDN southbound protocol for fine-grained wireless performance monitoring.

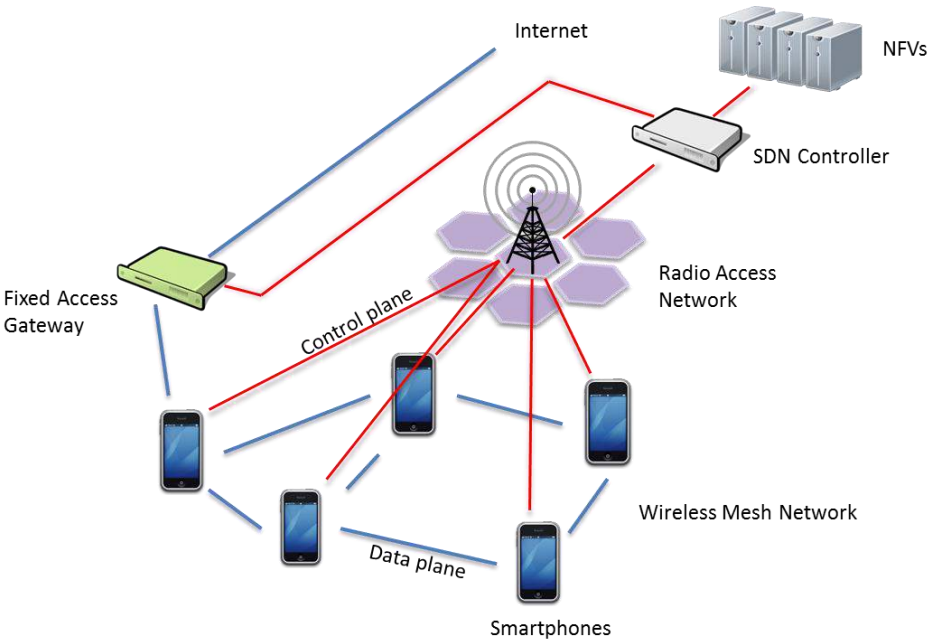


Figure 16 – Communication architecture for the SDN-controlled Wireless Mesh Network

As presented in **Figure 16**, the use case targets to demonstrate establishment and control of an IP wireless mesh network by the use of NFV supervisor (through SDN control plane). The selection of an alternative access network gateway (for instance a fixed access point – a home premise box) could be the root point for the creation of an IP wireless mesh network with neighbouring end-terminals.

Architecture design:

- Each smartphone is equipped with at least two wireless interfaces:
  - One cellular interface, used to communicate with the SDN controller (control plane).
  - One Wi-Fi interface, for data forwarding (data plane).
- Smartphones are configured to be SDN-enabled forwarding devices.
- The chosen protocol for routing in the mesh network is OpenFlow, being already widely used.
- All Wi-Fi interfaces of our mesh network (including the fixed access gateway) are in ad hoc mode. Indeed, using the capabilities of the ad hoc mode, it will allow the SDN controller to build an overall view of the network.

Operating principle:

Regarding the operating principle of the CEA solution, **Figure 17** summarizes the main steps. Indeed, offloading 3GPP Radio Access Network using SDN opportunities can be outlined as multiple interdependent services. First, we have a congestion monitoring service. That latter, has as main goal to estimate the state of charge of our overall mobile network and to detect incipient congestion state. Parallel to this, another service is responsible for building a global view of the network formed by all smartphones detected as well as the potential fixed access gateway. As soon as the state of congestion detected, we proceed to the verification of the existence of a potential mesh network that can be formed by smartphones. If effectively a potential mesh network is detected, the SDN controller disables the cellular mode of the smartphones and active the ad hoc mode. After this, the SDN controller installs the routing rules in the smartphones routing tables in order to reach the fixed access gateway. Once the mesh network formed, another service is responsible for monitoring the mesh network performances as well as ensuring mobility management of the smartphones. Below, each service will be described with more details.

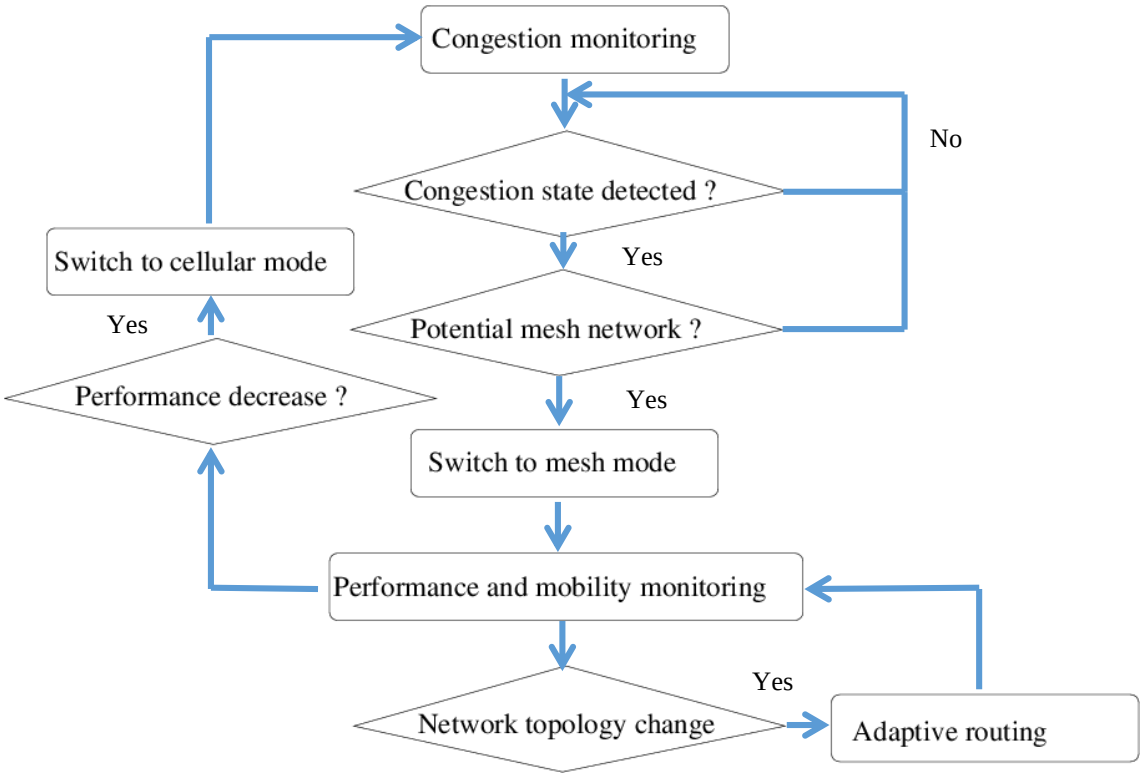


Figure 17 – General Structure for the operating principle

Congestion monitoring:

Basically, 3GPP Radio Access Network offloading operation should only be carried out when there is actual congestion on the network. Therefore, establishing of an accurate congestion detection mechanism is paramount to have an effective congestion management solution.

Nowadays, there is no standardized metric to detect congestion state. Multiple metrics have been proposed by researchers and companies. Among them, some are very complex to calculate, and others do not have a very accurate correlation with congestion.

Background:

Still in the context of network congestion management, [40] presents in their Whitepaper some relevant metrics and their effectiveness in congestion detection:

Detection Technique	Accuracy	Explanation
Time of Day	Extremely Low	The fundamental flaw of the time of day trigger is that it assumes congestion is occurring without actually confirming the case. As a result, management policies are implemented without technical justification (potentially in violation of best practices), are certainly not narrowly tailored, and are arguably out of proportion (to the nonexistent congestion).
Concurrent User Thresholds	Low	Measuring concurrent users at least relies upon measuring something, but it makes an assumption that the network is congested without actually verifying the case. In fact, the correlation between the number of users and the presence of congestion is extremely weak.
Bandwidth Thresholds	Medium	There is an assumption that bandwidth automatically causes a degradation in quality of experience. While overwhelming bandwidth does have a high correlation with

		lowering QoE, it is entirely possible for a link or resource to be at or close to maximum capacity without causing subscribers to suffer lowered QoE. Critically, though, there is an underlying assumption that link/resource capacity has a fixed maximum, but that is not the case either for mobile access networks or for fixed access networks. As a practical result, it is impossible to pick a threshold that is guaranteed to be below the point at which congestive collapse begins.
Subscriber Quality of Experience	Extremely High	The theoretically best way to trigger congestion management is with a direct measurement of congestion itself. The best thing is to measure a metric that is proven to have direct correlation with network congestion – ideally a metric that is caused by the congestion, rather than vice versa. Access round trip time (aRTT) is such a metric. Not only is aRTT known to increase dramatically when congestion is present, but it also has high correlation with subscriber assessments of quality of experience.

Figure 18 – Summary of Detection Techniques [40]

On the other hand, [41] have conducted studies for the case of user-plane congestion management in LTE EPS. To that end, they proposed the following congestion detection solution:

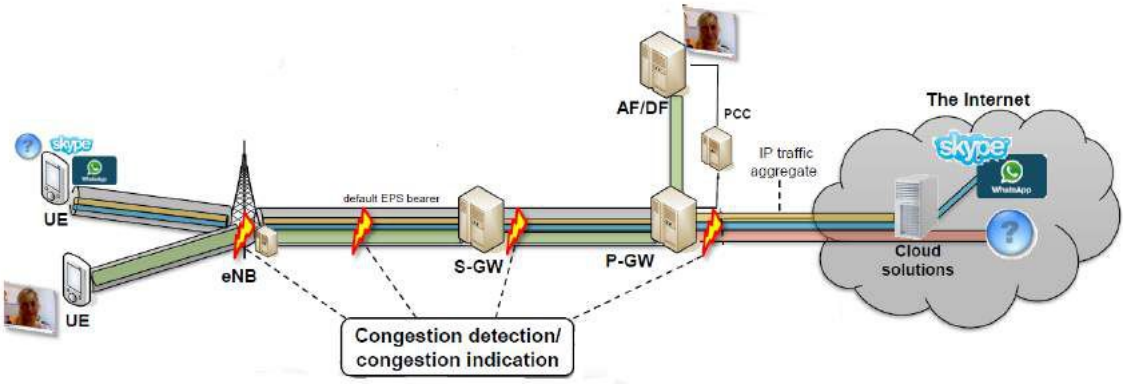


Figure 19 – congestion detection in LTE network

Aiming to have an exhaustive congestion detection system, monitoring agents should be placed in Radio Access, Backhaul or Core Network entities. Regarding the metric used in this work, the congestion detection is based on an extension to the IP and TCP, namely Explicit Congestion Notification (ECN) [42].

CEA congestion monitoring approach:

Below, we present the CEA solution concerning the congestion monitoring. First, regarding the metric used to detect the congestion state, the Bandwidth Thresholds seems to be the most adequate choice for our case. Indeed, developing an accurate congestion monitoring system is not the main aim here; therefore, measuring the bandwidth is both easier than aRTT and more effective than the other metrics. On the other hand, taking into account the solution advocated by [41] and thanks to the capabilities offered by the SDN approach, we can dynamically place monitoring agents in Radio Access, Backhaul or Core Network entities. **Figure 20** presents our approach interfaced with SIGMONA reference architecture.

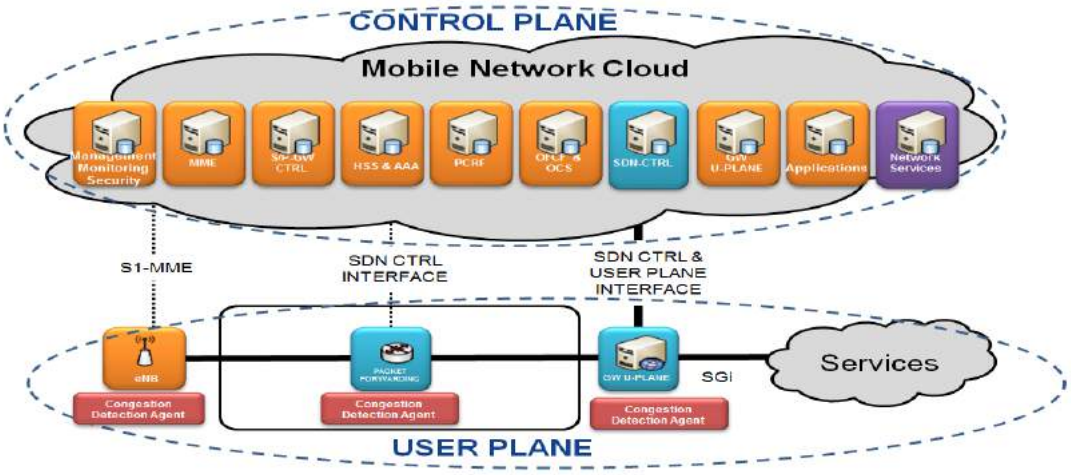


Figure 20 – CEA congestion detection in SIGMONA reference architecture

Conditions to switch in mesh mode:

Basically, the main criterion to switch in mesh mode is to be in the congestion situation. Indeed, the primary objective here is to investigate the opportunities offered by the SDN solution to offload 3GPP Radio Access Network. However, in addition to that, before offloading the LTE network, we must ensure that this operation is the most pertinent solution we have. Indeed, the metric used here for the congestion detection is the "Bandwidth Thresholds" which is based on assumptions about link/resource capacity. Therefore, we argue that before configure the smartphones in mesh mode, we must before verify that we'll get better performances. In fact, the performances of the mesh network relies on multiple metrics: the density and the mobility of the smartphones, the number of hops to reach the fixed access network, quality of WiFi links and so on. So, developing a strategy to estimate the performances of both cellular and mesh networks, figures among our research directions.

3.2.4.3 Main benefits, results & comparisons

This section presents results simulation carried out to investigate if the use of SDN in the mesh networks can help to improve the performances and this by mitigating or by overcoming the classical limitations known in these kind of networks. After an exhaustive survey about routing problems in WMNs, the scalability issue remainders the most common and important question to fix in all routing protocols. Aiming to deal with the scalability issue in wireless mesh networks, a large variety of simulations using different routing protocols (DSDV, OLSR, AODV, STATIC routing and SDN solution) has been conducted in order to show the ability of each of them to ensure performance in large-scale networks.

Figure 21 gives more details about the routing using our SDN approach. Each device (smartphone) informs the SDN controller about its neighborhood through "Hello messages". After this, the SDN controller is able to build the global topology of the network formed by the smartphones. As soon as a sender node does not find the mean to reach its target node, a request message is sent to the controller. Instantly, Dijkstra's algorithm is used to find the shortest path between source-destination nodes. Once this operation achieved, the controller installs the routing rules in all devices being part of the found path, enabling the sender node to attain its destination.

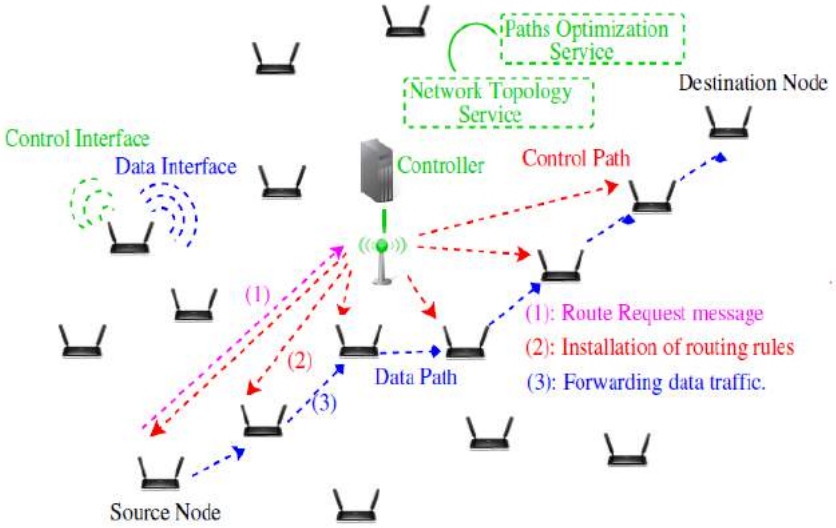


Figure 21 – Operating principle of the SDN solution

Among all the simulations conducted, we present the metrics that have a high correlation with subscriber assessments of quality of experience. Indeed, Figure 22 and Figure 23 compare the total lost packet rate and the throughput respectively (between all routing protocols simulated) while varying the number of hops between the device (smartphone) and the fixed access gateway. After examining the results, we can argue that the SDN solution can really improve the performances, and thus be more scalable compared to the conventional routing protocols: using SDN, (1) we mitigate interferences due to the signalling messages, (2) we reduce the convergence time of the routing process and (3) we ensure paths optimization.

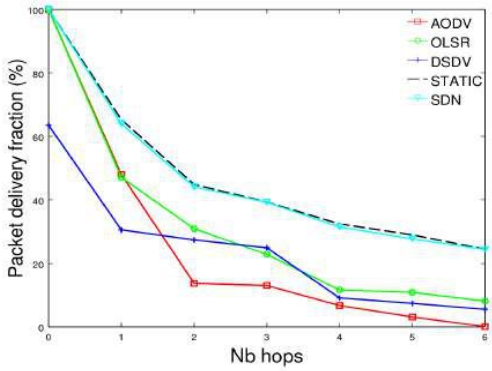


Figure 23 – Packet Delivery Fraction vs. Number of Hops

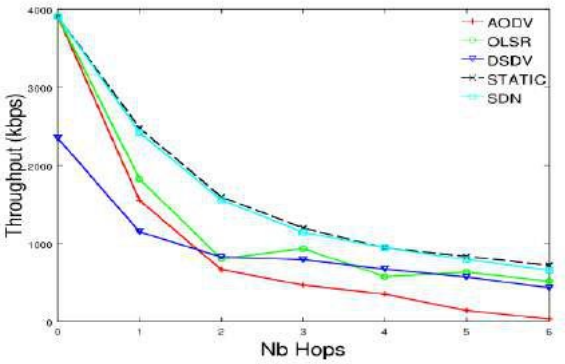


Figure 22 – Throughput vs. Number of Hops

3.3 Microscopic Traffic Management

3.3.1 DiffServ QoS architecture in SDMNs

3.3.1.1 Definition

DiffServ QoS architecture can provide soft or relative QoS guarantees in Layer-3 on top of different Layer-2 technologies in a scalable manner. Instead of flow-based QoS service provided by IntServ, DiffServ provides traffic class-based QoS for a small number of traffic classes. The IntServ model requires signaling towards each router in order to realize per-flow resource reservation. If the transport network supports QoS

services, then IntServ can provide hard QoS guarantees. In DiffServ model, the packets are policed (it is decided if the packet is in-profile, i.e, the given flow observes the SLAs) and marked at the edge SDN-switch. In-network gateways provide different scheduling (priorities) for in-profile and out-profile packets, based on the packet's traffic class, indicated by the marking.

3.3.1.2 Proposed solution and architecture

In software-defined mobile networks, we assume that a physical network infrastructure can be shared between multiple virtual networks at the same time. Several use cases can gain benefits from this use case, e.g., a mobile network operator could use virtual networks for service segregation or for selling transport network resources to other virtual mobile network operators. Figure 24 illustrates a DiffServ domain with three border gateways and one internal gateway, serving two L3 virtual networks. Each cercle denotes different IP domain per virtual/physical network.

Our objective is to provide appropriate level of QoS on Layer 3 for DiffServ traffic classes for each (physical and virtual) network transported over the same infrastructure. Our assumption is that L2 network resources are either overprovisioned or fit to the traffic demands of all virtual networks between L3 gateways. Layer-1/2 network elements are not illustrated in the figure.

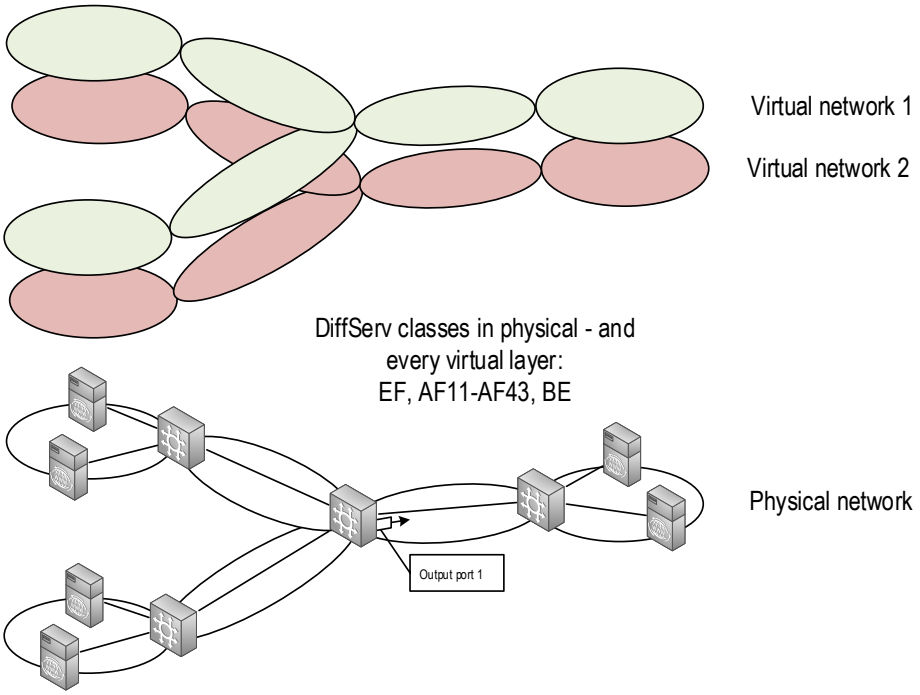


Figure 24 – Multiple L3 virtual networks shared on top of the same physical network.

For the control of DiffServ QoS settings, we propose an SDN enabler application, which integrates Routing and DiffServ control in L3. This application could be responsible for setting

- static routes per virtual/physical network;
- DSCP marking in border gateways, including traffic filters, DSCP value;
- OpenFlow meters for policing of DiffServ traffic classes for ingress traffic in border gateways, including meter type, maximum rate, drop precedence, traffic filter parameters;
- Queueing disciplines (qdiscs) and filters per virtual networks and traffic classes for shaping, including guaranteed, maximum rates, drop probabilities for Assured Forwarding classes and other queueing discipline parameters.

There are two essential ways of controlling DiffServ QoS settings:

- The APIs at the northbound interface of SDN controller and switch management protocols or commands enable to set the previous QoS settings and static routing rules. The necessary parameters must be derived in some way.

- The DiffServ QoS and routing application could interact with legacy PCRF over the Gx interface of PCRF in order to get appropriate policy control rules, which contain information on all the needed parameters.

3.3.1.2.1 Traffic shaping

Current OpenFlow and OF-Config specifications do not support the configuration of customized multi-level classful queueing disciplines, which are needed for appropriate scheduling of Diffserv traffic classes per virtual networks.

Appropriate traffic shaping requires a queueing discipline structure illustratred in Figure 25.

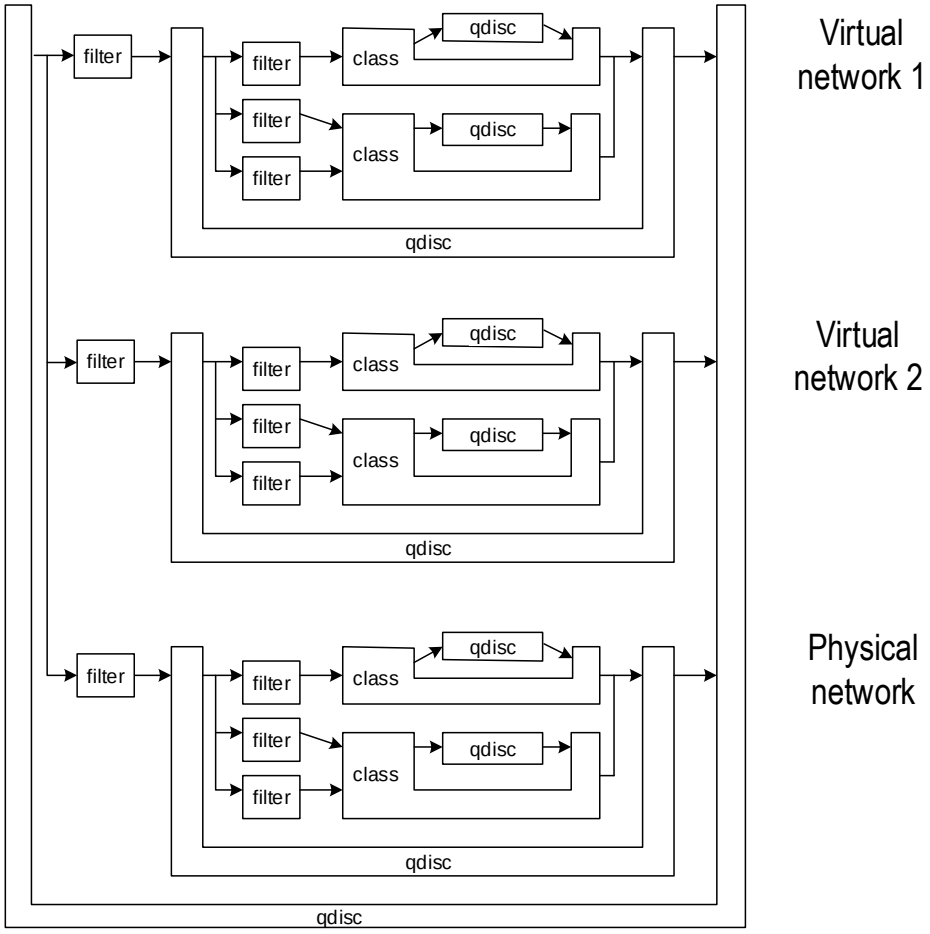


Figure 25 – Queueing discipline structure needed for DiffServ QoS in L3 transport network segments of SDMNs.

In this structure, the outermost qdisc (or the associated parent traffic class) determines the maximum rate for all traffic. The outermost qdisc should be classful in order to provide one traffic class per virtual/physical IP network. The filters of the outermost queueing discipline match on virtual network IDs or the absence of virtual network related headers of the packets. They map the traffic to a given class, which enforces guaranteed and maximum data rates for the aggregate traffic of a specific virtual network.

Under a virtual network-level traffic classes a queueing discipline is required, to enforce the QoS parameters of DiffServ traffic class. This is a classful queuing discipline, which enables to allocate guaranteed and maximum bitrates for each DiffServ traffic classes. Each traffic class is associated with one of the innermost queueing disciplines, which influences the scheduling properties of this traffic class. In our validations, we applied the following types of queuing disciplines.

- Hierarchical Token Bucket for the outermost, and virtual network-level queuing disciplines,
- Random Early Detection for the innermost queueing discipline of Best Effort traffic class
- Generalized Random Early Detection for Assured Forwarding Classes, including random early detection with different drop probability values per each drop precedence level,

- Packet limited First In, First Out (pfifo) queue for Expedited Forwarding with short buffer size.

Traffic shaping is required on each output port of each gateway in the DiffServ domain. It can be configured during the initiation phase of the networks. Different parameters of the classes and queueing disciplines can be changed, when the policy control rules are modified.

3.3.1.2.2 Traffic policing

Policing is about the formatting of ingress traffic according to the policy control rules described in SLAs. OpenFlow 1.3 sepcifies data rate meters, which aid in the adherence of the maximum rates of traffic classes. For charging, packet or data counters are more appropriate.

An OF rate meter is characterized by the meter ID, the maximum allowed data rate, the meter type, i.e., action to take for packets exessing the rate (drop or DSCP remark), and the precedence in case of DSCP remark type meters. It gives the amount by which the drop precedence level of packets excessing the allowed rate is increased. This is meaningful in case of Assured Forwarding traffic classes.

Policing is required only in border gateways.

3.3.1.2.3 Dynamic QoS control using PCRF

In this topic, we study the feasibility of an interface between legacy PCRF and SDN controller for dynamic QoS provision. This requires the modification of standard QoS policy enforcement, described in Appendix 2 of D3.1 [6].

3GPP Gx Diameter application defines two modes of policy control rule provisioning, i.e., solicited and unsolicited (by the PCEF).

In case of unsolicited PCC (Policy and Charging Control) rule provisioning the PCRF removes/deactivates old rules and installs/activates new rules due to

- an event scheduled internally by the PCRF, or
- an event caused by an OSS command, or
- an event caused by a Diameter Authentication-Authorization Request over the Rx interface.

In case of solicited PCC rule provisioning the PCEF demands modification of rules due to some event in the PCEF. This is a more complex scenario, which needs monitoring of events in the PCEF.

In SDN networks, the PCRF may communicate with the DiffServ QoS and Routing application of the SDN controller. Hence, this application should be able to provide the same control plane roles as the PCEF has for the 3GPP Gx Diameter application.

Figure 26 illustrates a possible way of dynamic policy control rule provisioning.
The main actors of this control scheme are

- OSS: it is responsible for managing the Subscription Profile Repository (SPR, i.e., subscriber database), e.g., adding/removing new subscribers, service types, binding services to subscribers. It manages the pre-defined policy control rules and the desicision logic for policy control in the PCRF.
- PRCF (SPR): this is the Policy and Charging Rules Function, which is connected with the SPR.
- Routing and DiffServ Application: SDN enabler for routing and DiffServ QoS control. Based on the PCC rules received from the PCRF via the Gx interface, it sets QoS parameters in SDN L3 gateways through Northbound API of the controller, or other APIs.
- SDN controller
- OF switches, in charge of L3 border or internal GW functions

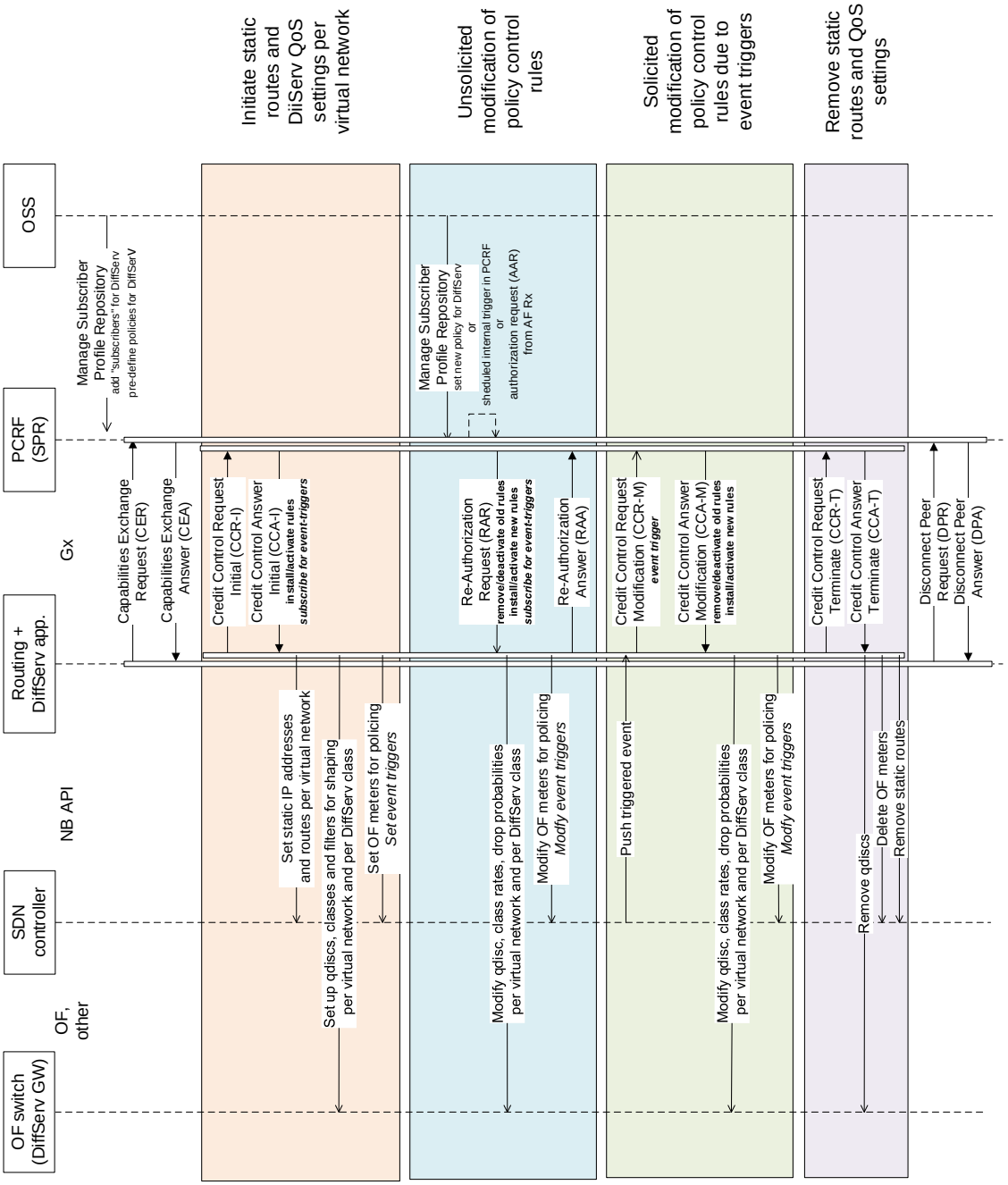


Figure 26 – Dynamic DiffServ QoS policy control rule provisioning from legacy PCRF to the Routing and DiffServ application of the SDN controller.

DiffServ QoS rule provisioning has the following main phases.

- When the Router and DiffServ control application is started, it initiates a Diameter connection with the PCRF. First a Diameter capabilities negotiation must be carried out, in order to determine what Diameter applications are supported by each peer. The SDN controller and PCRF exchange their supported Diameter applications using Capabilities Exchange Request and Answer (CER, CEA). In this scenario, both peers must support the 3GPP Gx application.
- Credit Control Request and Answer of type INITIAL_REQUEST (CCR-I, CCA-I) initiate a Diameter session, including policy control rules for all traffic types of the controller in CCA-I.
- Due to a scheduled change in the DiffServ policies at the PCRF, the PCRF initiates an unsolicited PCC rule modification after certain time with Re-Authorization-Request and Answer (RAR, RAA). The RAR command includes the rules to be removed/deactivated and installed/activated.
- If the Routing and DiffServ QoS application is capable of monitoring Event-Triggers, when an event is triggeded, the application will send a Credit Control Request and receive a Credit Control Answer

of type MODIFICATION (CCR-M, CCA-M). The CCR-M command includes the rules to be removed/deactivated and installed/activated.

- Finally, there is a Diameter session termination request, using Credit Control Request and Answer of type TERMINATION_REQUEST (CCR-T, CCA-T). This is triggered by the SDN controller. It automatically deletes the previously installed PCC rules at the SDN controller side, and the removes Diameter session at the PCRF side.
- When the PCRF or the Routing and DiffServ application wants to disconnect, they exchange a Disconnect Peer Request and Answer message.

In case of legacy Gx interface, one Diameter session—uniquely identified by Session-Id—refers to the policy control and charging of a given UE—identified by Subscription-Id. The Diameter session conveys policy control related information for all or a subset of traffic flows of a specific UE.

In our case, we apply one Diameter session between the SDN controller and the PCRF—identified by Session-Id. The Diameter session conveys policy control related information for all traffic types of the SDN controller, including every (V)LAN and every DiffServ class. We note that it would also be possible to apply finer granularity levels, e.g., to utilize one Diameter session per VLAN or even per DiffServ traffic class.

Our test scenario showed us that certain changes are required compared to standard Gx signalling:

- Flow-Information Attribute-Value Pair (AVP) could include an AVP for "Virtual-Network-Id", to match the VLAN/LAN. In our test scenario, the Charging-Rule-Name conveys this information, using the following naming convention: <VLAN_name>:<DiffServ_class>[:<policy_id>]
- Type-Of-Service AVP: in legacy Gx interface, it is a traffic filter parameter in the Flow-Information AVP. In our test scenario, border gateways extend the usage of Type-Of-Service AVP field to configure the DSCP value when marking packets, matched based on the Flow-Description AVP.

3.3.1.3 Main benefits, results & comparisons

In this project, we analyzed the feasibility dynamic DiffServ-based QoS control using legacy PCRF in virtual mobile networks. We analyzed two modes of PCC rule provision over the Gx interface, i.e., solicitated by the PCRF and unsolicitated. Our tests showed an example of Gx communication for the unsolicited PCC rule provisioning.

We could set the appropriate QoS settings at the output ports of specific OFsoft switches to enable DiffServ traffic classes per virtual networks. Current OpenFlow specifications do not allow these settings; hence we only could make the settings with Operating System level commands. We also showed that OF meters can be used for traffic policing in border gateways.

DiffServ may work in parallel with other components, which do not modify the routing and QoS settings in the DiffServ network domains, i.e., on the specific virtual transport network segment. Requests in regards of new policy control rules from other applications should be done over the Rx interface of the PCRF. PCRF should have the global vision on all QoS related requests and should keep track of QoS enforcement and monitoring.

Possible interworking scenarios are optimal video delivery, service chaining, mobility management scenarios,

3.3.1.4 Recognized issues and future work

DiffServ or any other IP based QoS marking does not ensure quality of the service or a specified service-level agreement (SLA). By marking the packets, the sender indicates that it wants the packets to be treated as a specific service, but it can only hope that this happens. It is up to all the service providers and the forwarding elements in the path to ensure that their policies will take care of the packets in an appropriate fashion. In order to achieve an end-to-end allocation of resources across separate domains, the Bandwidth Broker managing a domain will have to communicate with its adjacent peers, which allows end-to-end services to be constructed out of purely bilateral agreements. RFC 2638 from IETF defines the entity of the Bandwidth Broker in the framework of DiffServ. A Bandwidth Broker is an agent that has some knowledge of an organization's priorities and policies and allocates bandwidth with respect to those policies.

A recognized issue is that it still may not guarantee soft QoS guarantees over custom network topologies. For that, further optimization algorithms are needed, to translate QoS information in PCC rules to appropriate QoS settings on gateway level.

Another recognized issue is the lack of monitoring specific events in the SDN controller per (V)LANs and DiffServ traffic classes. This is required by solicited PCC rule provisioning. E.g., we should be able to monitor QoS rate excess events at the northbound interface of the SDN controller. Furthermore, freePCRF could provide very fine-grained and highly customizable decision logic. E.g., it can consider UE Subscribber Id, Radio Access Technology information, SGSN Id, etc. Certain part of this information is obtained through GTP-C in 3GPP networks. We should research appropriate ways to get those information e.g., through the OF switches and SDN controller or from the MME.

3.3.2 QoE/QoS enforcement in SDMNs

3.3.2.1 Definition

In order to achieve acceptable service quality, the broad spectrum of Internet services requires differentiated handling and forwarding of the respective traffic flows within increasingly "Internet Protocol (IP)" based mobile networks. The "3rd Generation Partnership Project (3GPP)" standard based procedures allow for such service differentiation by means of dedicated "GPRS Tunnelling Protocol (GTP)" tunnels, which need to be specifically setup and potentially updated as the mixture of client initiated service consumption changes. The ISAAR (Internet Service quality Assessment and Automatic Reaction) framework augments existing quality of service functions in mobile networks by flow based network centric quality of experience monitoring and enforcement functions. The ISAAR framework is meant to be an all in one solution for service quality monitoring and QoE enforcement within an operator network. It consist of three functional blocks the quality monitoring (QMON), the quality rules entity (QRULE) and the quality enforcement (QENF). The task of QMON is flow detection and assessment as well as monitoring and estimation of the Quality of Experience of the respected Internet services and is a part of Sigmona WP2. QRULEs tasks are creation of policy rules for each observed flow and the flow manipulation for the respective flows is handled by QENF, which is manly targeted in Sigmona WP3. The architecture and the function of the former version of ISAAR without SDN and NFV support are described in detail in the paper "ISAAR (Internet Service Quality Assessment and Automatic Reaction) a QoE Monitoring and Enforcement Framework for Internet Services in Mobile Networks" [2].

3.3.2.2 Proposed solution and architecture

SDN/NFV - ISAAR

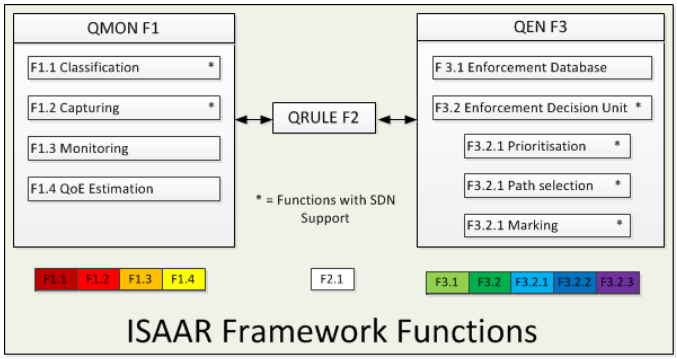


Figure 27 – ISAAR functional blocks with function split

Each of the frameworks functional blocks can be split in single functions as shown in Figure 27.

Until now, the deployment of ISAAR was either a distributed or a centralized approach. In the centralized version the only view monitoring and manipulation probes were deployed e.g. at the Gi/SGi interfaces of a mobile network. In the distributed approach many probes shall be deployed and spread all over the network, e.g. at Base Stations, NodeBs or eNodeBs. Both approaches have one thing in common on each observation/manipulation point a middle box has to be installed for monitoring and enforcement and each of thus boxes has to be able to carry out each of the functions mentioned in Figure 27.

One approach to overcome this limit is the utilization of Software Defined Networking (SDN) and Network Function Virtualization (NFV). The first step is the creation of an SDN only ISAAR, which is shown in Figure 28.

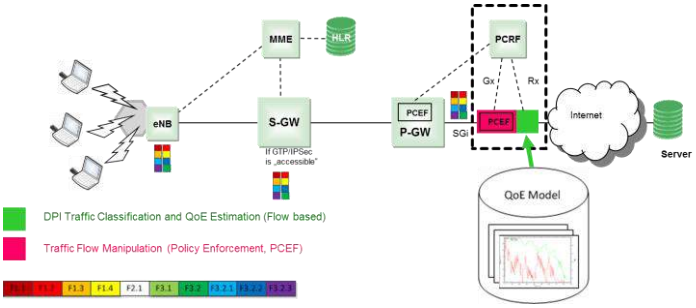


Figure 28 – SDN enhanced ISAAR

Two out of three functional blocks of ISAAR, the Quality Monitoring (QMON) and the Quality Enforcement (QENF) can benefit of using the functionalities of SDN. The quality monitoring of Internet services starts with identification of the respected flows within the traffic mix. Normally a lot of processing is needed for this detection, due to the Deep Package Inspection (DPI). To simplify that mechanism the SDN matching rules can be used. Therefore, the SDN controller has to be configured in a way, which allows identifying measurable traffic by using the matching rules and teeing it out for to one of the measuring points of ISAAR. The next advantage, which can be derived, is the real time tee out of regarded traffic. In most SDN implementations there are all mechanism which are needed to copy the measured traffic to an own port or switch a path to a measuring probe. Enabling the usage of viewer, more centralized probes.

On the other side SDN functionalities can be used for changing the per hop behaviour (PHB) for each traffic flow. If the quality is changing, ISAAR can signal the change in the PHB to the controller. Only the change, increasing or decreasing the priority of the flow, will be signalled the SDN controller decides on its own how the change in priority is done. Meaning the SDN controller becomes the QENF of ISAAR. Hence, it could be possible to monitor a certain stream, calculate the QoE of the service and react on the estimated value. If the service has a high MOS then ISAAR can signal the SDN controller to reduce the priority of the respected flow. If there are no other enforcement mechanisms within this network, the QENF functional block of the framework will be completely replaced by SDN.

However, the middle boxes still need to do every function in the place where they are deployed. Here the NFV is taken into account. Some of the function in the ISAAR framework can be virtualized and placed in a datacentre. For example the whole QRULE functional block can be realized in software and put anywhere in the network only the quality information of the observed flows has to be signalled to the rules entity and the designed behaviour has to be sent to the QENF functions. The quality monitoring, the QoE estimation, the enforcement database and the enforcement decision unit can be virtualized and placed in a datacentre, too. But, there are still functions which have to stay within the user data path.

3.3.3 Quality of Service Control and Resource Prioritization with SDN

3.3.3.1 Definition

We analyze how SDN can help to ensure a high-quality uninterrupted VoIP service accompanied by video in an emergency. On the other hand, there will also be regular users trying to communicate with one another in a network congested with background traffic, which is created by sources other than the VoIP service. These users should also receive some QoS, although less than premium, so that the impact of background traffic may be limited.

3.3.3.2 Background

In [4], the authors propose novel QoS extensions to distributed control plane architectures for multimedia delivery over large-scale, multi-operator SDNs. It provides topology aggregation and link summarization methods, an optimization framework for flow-based end-to-end QoS provision, and two scalable distributed secure inter-domain QoS routing control plane designs. In [14], the authors present an SDN-enabled convergent network for efficient resource management, E2E QoS enforcement, and flexibility and scalability for future network evolution to benefit from the SDN and converged network techniques. Similar to our experiments, they implemented some promising applications of the proposed network and evaluated based on an open test platform to show its superiority to current network architecture. [15] gives integrated traffic control structures for SDN while providing traffic engineering (TE) and fast failure recovery (FFR) features for class-of-service (CoS)-aware flow aggregates.

In [16], the authors propose an SDN-enabled capacity-sharing framework for User-Centric Networks (UCNs) in order to mitigate the problem of sharing limited network capacity and resources efficiently and fairly by taking into account current network load and conditions, and QoS requirements. Likewise, another use of a different network domain is in [17]. It extends this context-aware composition and delivery concept to utilize Next Generation Service Overlay Networks (NGSONs [18]) over SDN. Similarly, [19] combines SDN with Autonomic Network technologies to construct a software defined self-managing solution for the future network.

3.3.3.3 Proposed solution and architecture

Providing QoS in such scenarios with the help of SDN is very feasible, via limiting bandwidth, assigning flows to different queues and/or adapting routing decisions based on network conditions. In this experiment, Argela SDN Controller, so called *yakamOS*, is utilized to manage QoS for different flows, which communicates with Argela SIP Server for receiving Service Level Agreement (SLA) parameters per VoIP subscriber and making QoS decisions based on these parameters.

Argela SIP Server that is utilized during the tests is a development version of the Wirofon Server used in Turk Telekom's network for giving VoIP service to its subscribers. Hence, apart from the emergency case, this experiment has another important business potential: Transferring Wirofon into an SDN-enabled VoIP service for easy resource and QoS management. The following stakeholders are involved in the experiment: Internet Service Provider (ISP), Turk Telekom as VoIP Provider and VoIP Users (premium and regular).

In order to push the SDN test-bed (Figure 29) to its limits and make tests under extraordinary conditions, we have gone far more than Wirofon's daily usage. However, since we have used SIPp, which is a popular SIP testing tool for generating SIP traffic, it is possible to run the tests with less demanding parameters to realistically simulate the bandwidth usage of Turk Telekom's VoIP solution. In addition, SDN enables fine granular control over network traffic, where it is possible to classify and prioritize VoIP data (audio and video separately). Hence, Required Extra Capacity (REC) for VoIP is kept at minimal, if not zero. It is possible to differentiate audio and video flows of a VoIP session with SDN, via the source and destination ports of the connection. That is because in VoIP service, audio and video data are sent using different transport layer ports.

Determination of the ports that are to be used can be achieved by two methods: The SDN Controller can communicate with the SIP server and learn which ports are to be used for audio and video, or the SDN Controller can capture SIP signaling traffic and parse “SIP INVITE” and “SIP 200 OK” messages to get the port information. Once emergency service officers (i.e., first responders) are classified as premium users, it is possible to prioritize their traffic via bandwidth management, priority queuing and adaptive routing. In the results given in Section 0, it is evident that the premium VoIP traffic can be prioritized over background traffic and regular VoIP traffic. It is also possible to prioritize VoIP audio flows over VoIP video.

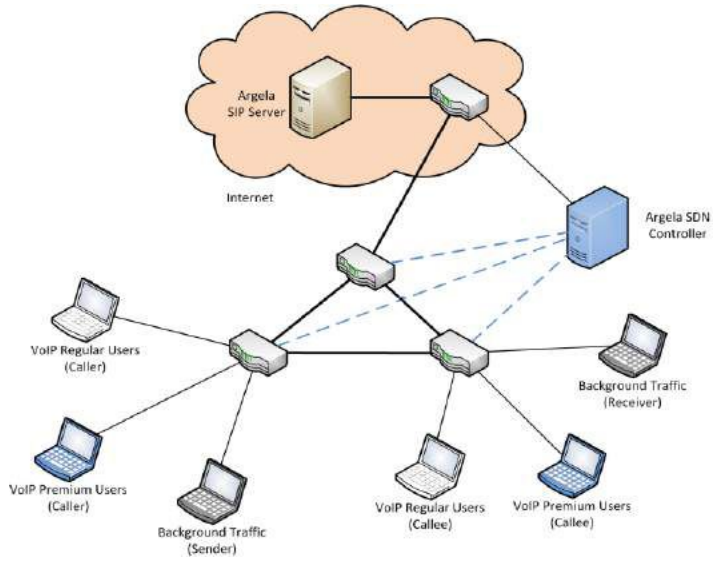


Figure 29 – Architecture for SIP-QoS

We defined three metrics that are commonly used in the literature to measure the QoS, these metrics are namely; loss rate, latency and jitter. The calculations of these metrics are based on the following model. Assume the sender side sends *N* packets to the receiver side, and i^{th} packet is transmitted at time t_{si} and received at the receiver at t_{ri} where $t_{ri} > ts_i$. Ignoring Operating System (OS) and framing delays by assuming they are slightly same at both ends, we can calculate the latency Δ_i for i^{th} packet as,

$$\Delta_i = t_{r_i} - t_{s_i}$$

Jitter, denoted by δ, is known as the deviation of the latencies; hence, we used the average absolute value deviation from the mean of the latencies. The mean latency μ_Δ is calculated as,

$$\mu_\Delta = \frac{1}{\|C\|} \sum_{\forall i \in C} \Delta_i$$

where *C* denotes the set of indices for correctly transmitted packets and $\|C\|$ is the cardinality of this set. Hence, lost packets are not being taken into account. Consequently, the jitter is calculated as follows,

$$\delta = \frac{1}{\|C\|} \sum_{\forall i \in C} |\Delta_i - \mu_\Delta|$$

Finally, the loss rate is calculated as the percentage of the lost packets. Assume *M* out of the *N* sent packets are transmitted correctly where $M = \|C\|$. Hence, *N-M* packets are lost in the transmission phase, then the lost rate Λ is straightforward to compute as,

$$\Lambda = \frac{N - M}{N}$$

where $N \neq 0$, later Λ is converted into percentage format for the sake of simplicity in the evaluation phase.

3.3.3.4 Main benefits, results & comparisons

In this study, we utilized the use of SDN to provide a guaranteed QoS to premium users in a VoIP system congested by both from VoIP and non-VoIP background traffic by limiting bandwidth, assigning flows to different queues and adapting routing decisions based on network conditions. In addition, audio traffic is efficiently prioritized over video in order to keep communication reliable for heavy-loaded networks. Results revealed that the flexibility enabled by SDN improved the QoS in the network with a reduction in loss rate more than 5% and more than 50% of reduction in latency and jitter. Accordingly, we can conclude that this enhancement can be further applied to other networks and applications as well.

3.3.3.5 Recognized issues and future work

Problems with the SIPp program and the development version of the SIP server created problems while testing the framework. Also, for evaluating the QoS metrics, jNetPcap was used which has problems with the SIP protocol.

As a future work, we will integrate into this framework the traffic and resource management algorithms developed as part of the WP3 project to further enhance QoS paremeters.

4. Architectural Modifications, Integration Points and Possible Performance Improvements

4.1 Macroscopic Traffic Management

4.1.1 Coordinated traffic and resource management and efficient routing in SDMNs

4.1.1.1 Updates on architecture mapping

The coordinated traffic and resource managemenet module uses the following components in the SDN-layered reference architecture model, depicted in Figure 30. The main required functions are as follows:

1. OFConfig/OVSDB capable OF switches
2. Flow programming and flow statistics collection
3. Optimal routing application
4. Switch management with OFConfig/OVSDB
5. Resource monitoring for optimal routing calculation

The required interfaces are as follows:

- 1-2: OF: FlowMod, PacketIn, …., Packet statistics
- 1-4: OFConfig/OVSDB configuration
- 1-5: Switch status
- 2-3: Optimal routing guidance to
- 2-5: Flow stats to monitoring unit
- 3-4: Optimal routing guidance to switch management for OFConfig/OVSDB configuration on switch
- 3-5: All monitoring info collected by optimal routing app from monitoring

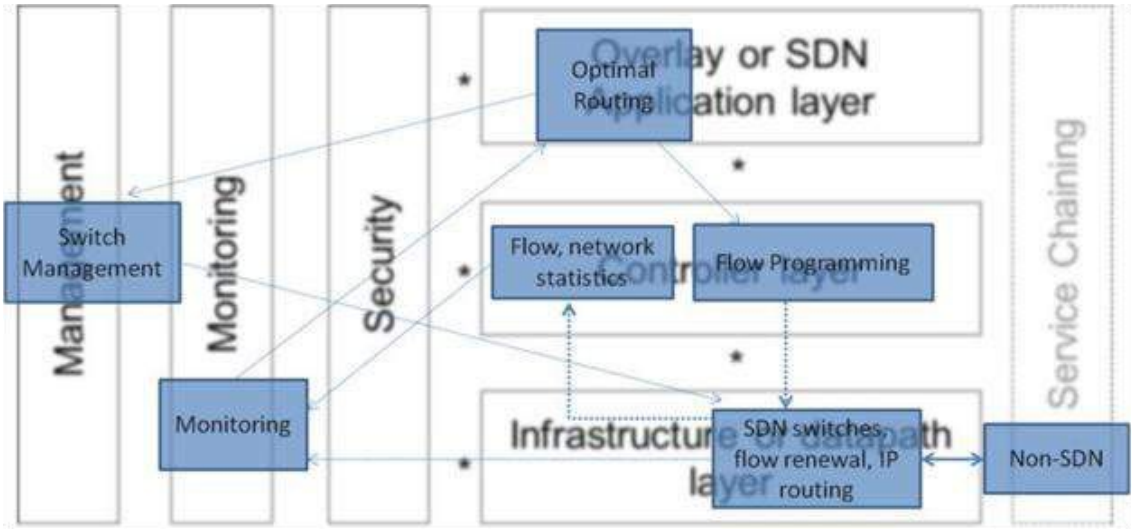


Figure 30 – Coordinated traffic and resource management (SDN-layered reference model view)

The changes to the reference SIGMONA architecture can be seen in Figure 31. The optimal routing application sits on top of the controller and communicates with the controller via the North Bound Interface API. The controller supplies the topology and the capacity of the links to the optimal routing application for making routing decisions. The monitoring module supplies the current state of the network so the optimal routing application can make routing decisions and relay this to the controller. The controller then sends OpenFlow packets to the switches to setup the paths. The optimal routing application also uses the management interface to turn off switches (or ports) according to the requirements of the network. If the load is low, the optimal routing application commands the management application to turn off ports and switches to use the power efficiently.

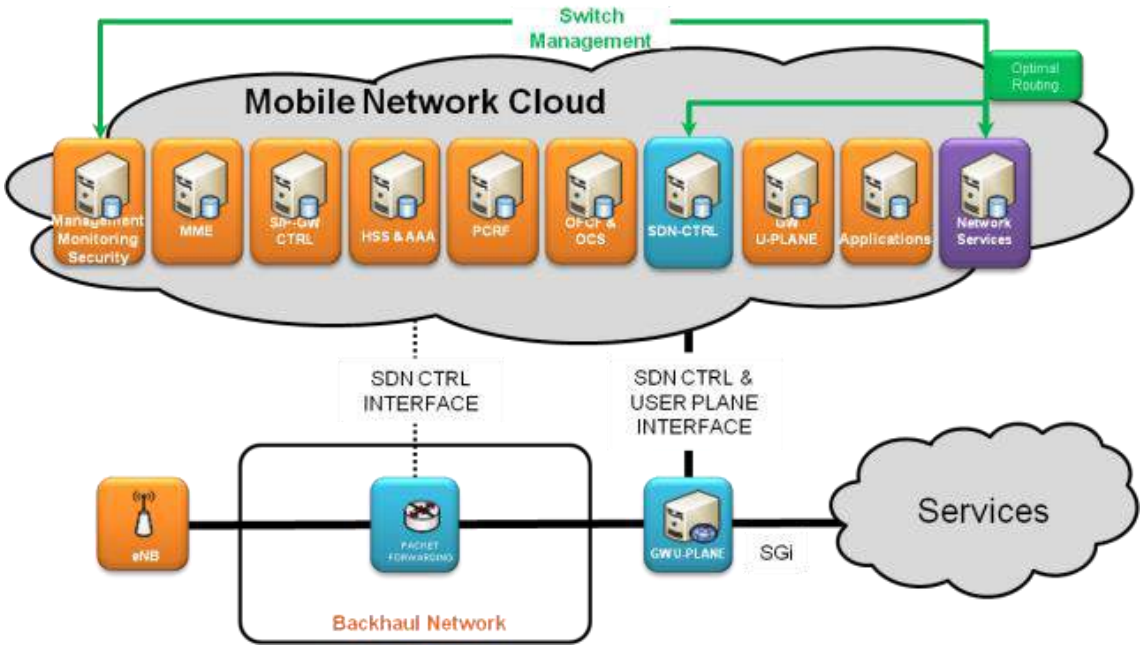
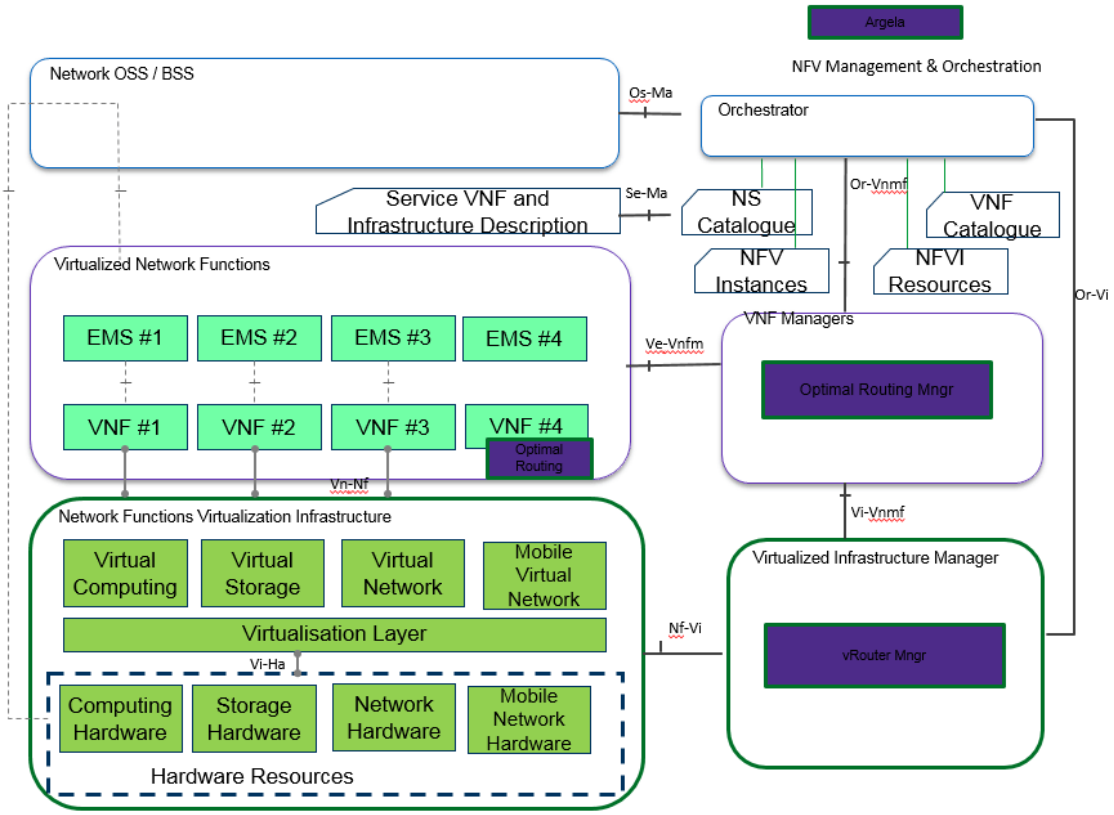


Figure 31 – Coordinated traffic and resource management in SIGMONA reference architecture

The reference architecture mapping, which based on ETSI NFV ISG, has shown in SIGMONA D1.2.



Argela mapping to ETSI NFV architectural framework is listed below:

- Optimal Routing VNF: The Optimal routing application in this architecture works as VNF. For the bare-metal case, the Optimal Routing application communicates with the ***OF Controller*** through OF APIs for routing, switch and port enabling/disabling, statistics and configuration purposes. The information coming from the network is relayed to the Optimal Routing application and the application decides the routing rules and configuration changes in the network.

- Optimal Routing Manager: In the case where there are virtual switches in the network, disabling/enabling a port/switch might also mean to create a new VM on a server or disabling the VM. In this case, this decision from the Optimal Routing VNF should reach the VIM and this communication is carried out through the Optimal Routing Manager that is one of the VNF managers. The Optimal Routing Manager then coordinates these requests with the vRouter Manager in the VIM (working with OpenStack or equivalent).

- vRouter Manager: When a request for instantiating/deinstantiating a VM is requested from the Optimal Routing Manager according to the directions of the Optimal Routing VNF, this module creates and makes ready (or deletes) the VM for carrying out the defined vSwitch duty on the node specified.

4.1.1.2 Detailed interface descriptions

The optimal routing application communicates with the controller via the northbound API using JSON objects. Initially, the controller provides the optimal routing application the topology and the capacities:


```
{
 "Switches":[
 {
 "mac":"32:00:00:00:32:16",
 "ipv4":"192.168.50.22"
 },
 {
 "mac":"32:00:00:00:32:15",
 "ipv4":"192.168.50.21"
 },
 {
 "mac":"32:00:00:00:32:0B",
 "ipv4":"192.168.50.11"
 },
 {
 "mac":"00:9C:02:B8:FF:C0",
 "ipv4":"192.168.50.41"
 }
 ],
 "Links":[
 {
 "srcmac":"32:00:00:00:32:0B",
 "srcport":3,
 "destmac":"32:00:00:00:32:15",
 "destport":3,
 "capacity":500
 },
 {
 "srcmac":"32:00:00:00:32:15",
 "srcport":2,
 "destmac":"32:00:00:00:32:16",
 "destport":2,
 "capacity":300
 },
 {
 "srcmac":"32:00:00:00:32:0B",
 "srcport":5,
```

```
"destmac":"32:00:00:00:32:16",
"destport":5,
"capacity":200
},
{
"srcmac":"00:9C:02:B8:FF:C0",
"srcport":7,
"destmac":"32:00:00:00:32:16",
"destport":3,
"capacity":1000
}
],
"Hosts":[
{
"mac":"00:17:A4:EA:D5:D7",
"ipv4":"192.168.50.202",
"switchmac":"32:00:00:00:32:0B",
"switchport":11
},
{
"mac":"00:17:A4:EA:D5:D5",
"ipv4":"192.168.50.200",
"switchmac":"00:9C:02:B8:FF:C0",
"switchport":9
}
]
}
```


The controller can then request a path from the optimal routing application with the following JSON:


```
{
"srcipv4": "192.168.50.200",
"dstipv4": "192.168.50.202",
"rate": 300
}
```


And the optimal routing algorithm provides the following path:


```
{
"Path":[
{
"srcmac":"00:9C:02:B8:FF:C0",
"srcport":7,
"destmac":"32:00:00:00:32:16",
"destport":3
},
{
"srcmac":"32:00:00:00:32:16",
"srcport":2,
"dstmac":"32:00:00:00:32:15",
"dstport":2
},
{
"srcmac":"32:00:00:00:32:15",
"srcport":3,
"dstmac":"32:00:00:00:32:0B",
"dstport":3
}
]
}
```

The optimal routing application can also turn off switches (or ports) by sending the following message to the management application:

```
{
 "switchmac": "32:00:00:00:32:16",
 "switchport": "5"
}
```

By setting the port number to -1, the optimal routing application can turn off the whole switch.

4.1.1.3 Performance improvements (reactive vs. proactive, etc.)

4.1.2 Application-Layer Traffic Optimization techniques in SDMNs

4.1.2.1 Updates on architecture mapping

Sections 3.2.2.2.2 and 3.2.2.2.3 describe the main functional elements of ALTO server and ALTO client. Here we present the possible placement of these functions in the SIGMONA reference architecture figure.

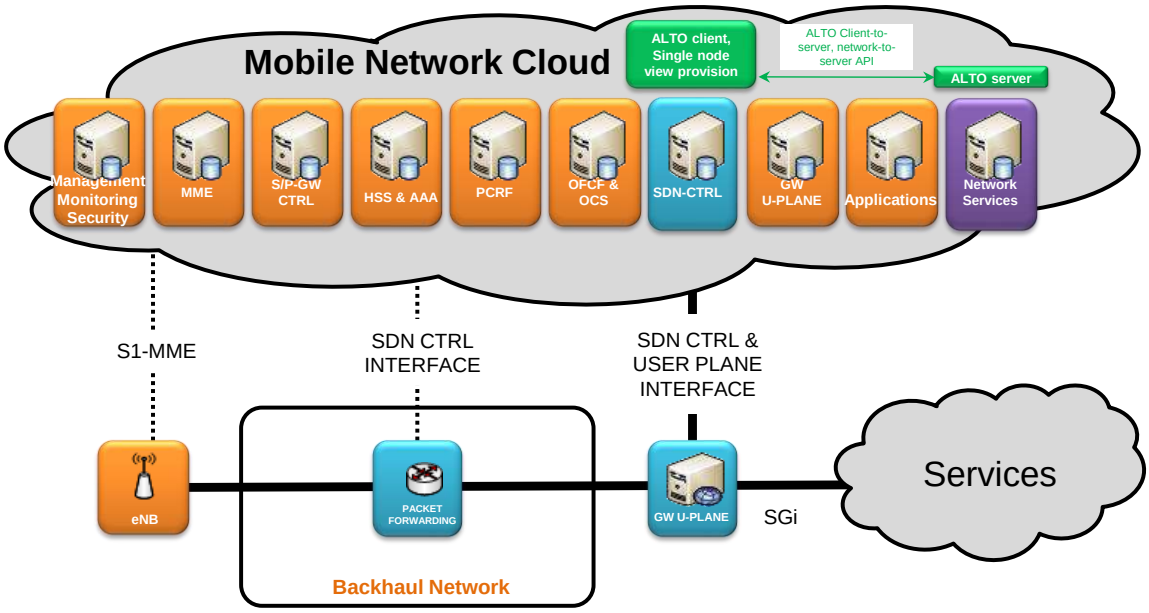


Figure 32 – ALTO-SDN in SIGMONA reference architecture.

Mobile Core Network Cloud is based on architecture framework defined in ETSI NFV Architectural Framework [20], Section 5.2 “High-Level NFV Framework”. Some of the functional blocks illustrated within Mobile Core Network Cloud such as Mobility Management Entity (MME), Policy Charging and Control Rules Function (PCRF) can be seen as Virtualised Network Functions (VNFs). In the Mobile Core Network Cloud deployment, logical layers such as Management, Monitoring, Security, Overlay/SDN Application Layer, Controller Layer, virtualized Infrastructure/Datapath Layer and Service Chaining may co-exist simultaneously.

ALTO SDN requires the following modifications: i.e., additional components in this architecture.

- SDN controller: ALTO client requesting ALTO network and costmaps, and single network view providing network and cost maps of the controlled SDN area are additional extensions
- Network services: ALTO server, but in case of inter-domain service, it may also be operated by another operator

Additionally, ALTO provides rendezvous services for distributed services; therefore, CDN servers/caches are present in the service cloud.

Required interfaces:

- ALTO Client-to-server REST API (RFC 6708)

- ALTO Network-to-server REST API (single node view provision)

In addition to this mobile network cloud view, we also present the required components in the layered SDN reference model, based on ONF definition [21]. Based on this definition, potential architecture layers for components are

1. Infrastructure or datapath layer
2. Controller layer
3. Overlay or SDN application layer
4. Management
5. Monitoring
6. Security

Potential reference points are

1-2: infrastructure or datapath layer – controller layer

2-3: controller layer – overlay or SDN application layer

1-4 / 1-5 / 1-6: infrastructure layer – management / monitoring / security

2-4 / 2-5 / 2-6: controller layer – management / monitoring / security

3-4 / 3-5 / 3-6: overlay or SDN application layer – management / monitoring / security

ALTO-SDN use case has the following components in the SDN-layered reference architecture model.

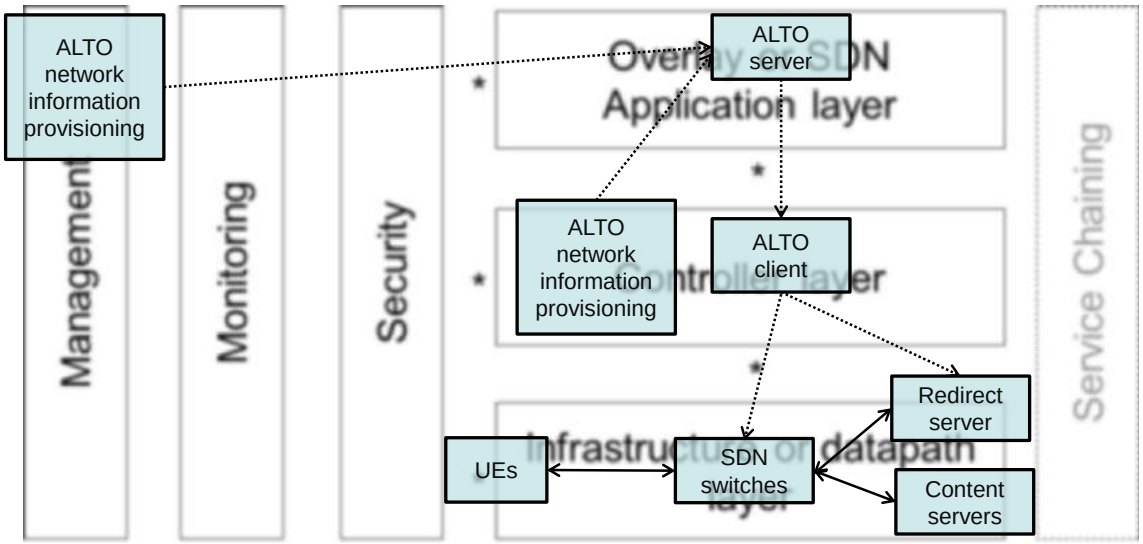


Figure 33 – ALTO-SDN service (SDN-layered reference model view).

The main additional functions in the SDN architecture layers are the following:

1. Controller-layer: ALTO client, ALTO network information provision from SDN domain
2. SDN Application-layer: ALTO server (also could be placed in management-layer)
3. Management-layer: ALTO network information provisioning

Required reference points are the following:

Ref. point '1-2': OpenFlow protocol, e.g., FlowMod, PacketIn

Ref. point '2-3': ALTO network and cost map queries, ALTO network information provision

Ref. point '3-4': further network information provision, e.g., from CDN operators (not implemented).

4.1.2.2 Detailed interface descriptions

4.1.2.2.1 ALTO Client-to-Server interface

The ALTO client and server communicate with specific HTTP requests and responses over REST API. The ALTO protocol specifies several JSON media types for denoting different types of messages. These are summarized in Table 3.

Table 3 – JSON media types specified by the ALTO protocol [1].

Type	Subtype
application	**alto-directory+json**
application	**alto-networkmap+json**
application	alto-networkmapfilter+json
application	**alto-costmap+json**
application	alto-costmapfilter+json
application	alto-endpiontprop+json
application	alto-endpointpropparams+json
application	alto-endpointcost+json
application	alto-endpointcostparams+json
application	**alto-error+json**

4.1.2.2.2 ALTO Server-to-Network interface

The SDN controller provides an up-to-date single-node network view over RESTful interface with JSON media type. An example for the JSON message is given in the following:

```
{"topology":{
"10.0.0.1":{"10.0.0.2":{"num-routing":2, "num-delay":0},"10.0.0.3":{"numrouting":
6, "num-delay":0},"10.0.0.4":{"num-routing":6, "num-delay":0}},
"10.0.0.2":{"10.0.0.1":{"num-routing":2, "num-delay":0},"10.0.0.3":{"numrouting":
6, "num-delay":0},"10.0.0.4":{"num-routing":6, "num-delay":0}},
"10.0.0.3":{"10.0.0.1":{"num-routing":6, "num-delay":0},"10.0.0.2":{"numrouting":
6, "num-delay":0},"10.0.0.4":{"num-routing":2, "num-delay":0}},
"10.0.0.4":{"10.0.0.1":{"num-routing":6, "num-delay":0},"10.0.0.2":{"numrouting":
6, "num-delay":0},"10.0.0.3":{"num-routing":2, "num-delay":0}}},
"pidMask":"255.255.255.255",
"mapName":"my-default-network-map"}
```

The proposed structure is similar to the ALTO network map; it defines abstracted one-way links between subnets, which will be assigned to PIDs by the ALTO server. The network mask for the subnets is given by 'pidMask'. The 'mapName' gives the network map and associated cost maps which should be updated.

4.1.2.2.3 ALTO Client-to-Redirect Server interface

In case of non-transparent, HTTP redirect server based redirection; the redirect server must get the information on the preferred endpoint for a given incoming TCP connection. We defined a HTTP POST message between the ALTO client and the HTTP redirect server to convey this information using JSON format, as illustrated in the following.

```
{"src":{"10.0.0.1", 12345}, "dst":{"10.0.0.3", 8008},"redir":{"10.0.0.2", 8000}}
```

The message contains the IP address and port of the source, initial destination and preferred destination (redir).

4.1.2.3 Performance improvements (reactive vs. proactive, etc.)

4.1.3 Joint traffic and cloud resource management for service chaining in SDMNs

4.1.3.1 Updates on architecture mapping

The Joint traffic and cloud resource management for service chaining in SDMNs controller module uses the following components in the SDN-layered reference architecture model, depicted in Figure 34. The main required functions are as follows:

6. OpenFlow and OVSDB capable switches
7. Flow and switch statistics collection
8. Service chaining application (SCO)
9. Flow programming with OpenFlow
10. Resource/Inventory monitoring (Managament and Administration Layer) for service chaining

These functions perform the following tasks:

- 1-2-3: Packet, flow and switch statistics unit (monitoring)
- 1-3-4: Switch manager and forwarding rules manager for flow programing
- 3-5: Resource status (monitoring) and inventory management unit to create chains
- 2-4-5: Optimal routing and traffic management unit

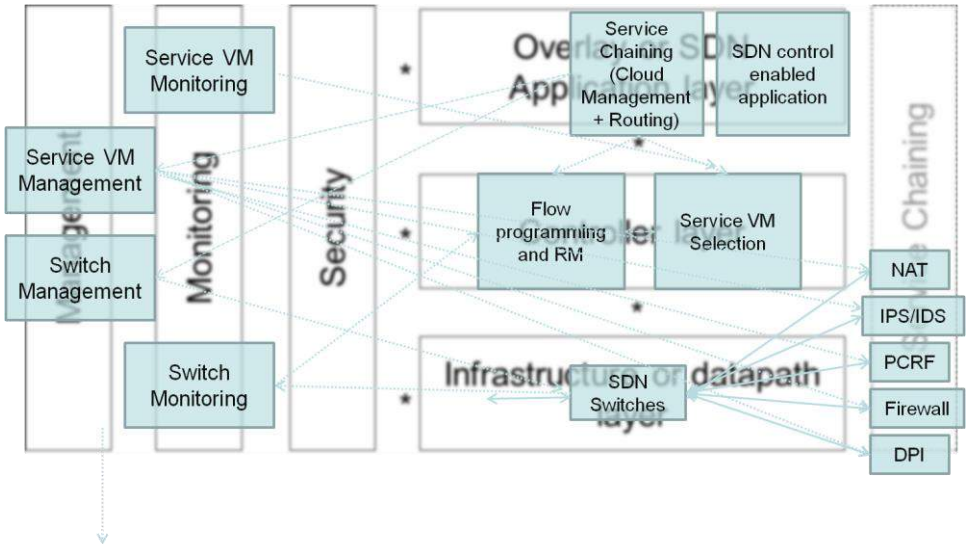


Figure 34 – Architectural Model for Service Chaining optimization in SDMNs

The changes to the reference SIGMONA architecture can be seen in Figure 35. The service chaining application sits on top of the controller and communicates with the controller via the North Bound Interface API and controller API.

The controller supplies the topology, capacity and statistics of the links. The Management & Administration Plane supports interfaces for node and network management. These interfaces are used to collect inventory, topology and statistics information about SDN, non-SDN network and other service function nodes.

To configure service chain path, service chaining application needs to communicate with this layer to get information. This information is used by service chaining application to create service chain sets and make optimal routing decisions. Management & Administration Plane can be a Network Management System (NMS) or a Cloud Orchestration Layer.

The main task of service chaining application is creating chain paths and setting flow rules to SDNC via North Bound Interface API and controller API. SDNC translates these to Match and Action rules which are entered to the forwarding elements. By doing this, forwarding elements forward the packets according to these match-action lists. By using the information which is provided by NMS, service chaining application can configure and reconfigure flow rules.

Required interfaces:

- Service Chaining Application – to – Management & Administration Plane API
- Service Chaining Application – to – controller API

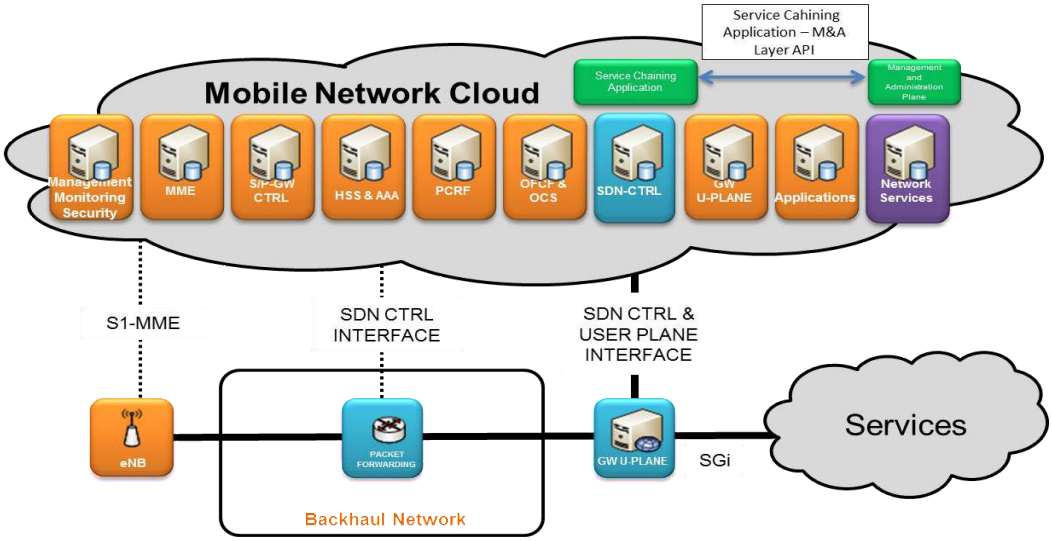


Figure 35 – Service Chaining in SIGMONA reference architecture.

4.1.3.2 Detailed interface descriptions

The service chaining application communicates with the controller via the controller API. Initially, the controller provides topology, capacities, switch management, statistic management and forwarding rules management (flow rules).

Since OpenDaylight based OSGI framework provides modular system, one can install their own application (bundles) without interrupting the overall system.

```
osgi> ss
.
.
251 ACTIVE org.opendaylight.controller.switchmanager.implementation_0.4.2.SNAPSHOT
252 ACTIVE org.opendaylight.controller.sal-inmemory-datastore_1.1.0.SNAPSHOT
254 ACTIVE org.opendaylight.controller.networkconfig.neutron.implementation_0.4.2.SNAPSHOT
256 ACTIVE org.opendaylight.controller.sal-compatibility_1.1.0.SNAPSHOT
257 ACTIVE org.opendaylight.controller.md.inventory-manager_1.1.0.SNAPSHOT
258 ACTIVE org.opendaylight.controller.hosttracker.implementation_0.5.2.SNAPSHOT
259 ACTIVE org.opendaylight.controller.sal-rest-connector_1.1.0.SNAPSHOT
261 ACTIVE org.opendaylight.controller.usermanager.implementation_0.4.2.SNAPSHOT
262 ACTIVE org.opendaylight.controller.netty-config-api_0.2.5.SNAPSHOT
263 ACTIVE com.fasterxml.jackson.core.jackson-databind_2.3.2
.
.
281 INSTALLED com.ericsson.sco_app_0.0.1
osgi> start 281
```

Controller APIs:

```
IForwardingRulesManager frm;
ITopologyManager topologyManager;
ISwitchManager switchManager;
IStatisticsManager statisticsManager;
```

```
IForwardingRulesManager forwardingRulesManager;
NodeConnectorStatistics ncStats;
```

Get topology information:

```
topologyManager.getNodeEdges();
{
OF|00: 00: 00: 00: 00: 00: 00: 02=[(OF|1@OF|00: 00: 00: 00: 00: 00: 00: 02-
>OF|4@OF|00: 00: 00: 00: 00: 00: 00: 01),(OF|4@OF|00: 00: 00: 00: 00: 00: 00:
05->OF|2@OF|00: 00: 00: 00: 00: 00: 00: 02),(OF|2@OF|00: 00: 00: 00: 00: 00:
00: 02->OF|4@OF|00: 00: 00: 00: 00: 00: 00: 05),(OF|4@OF|00: 00: 00: 00: 00:
00: 00: 01->OF|1@OF|00: 00: 00: 00: 00: 00: 00: 02)],

OF|00: 00: 00: 00: 00: 00: 00: 03=[(OF|1@OF|00: 00: 00: 00: 00: 00: 00: 04-
>OF|2@OF|00: 00: 00: 00: 00: 00: 00: 03),(OF|2@OF|00: 00: 00: 00: 00: 00: 00:
03->OF|1@OF|00: 00: 00: 00: 00: 00: 00: 04),(OF|1@OF|00: 00: 00: 00: 00: 00:
00: 03->OF|5@OF|00: 00: 00: 00: 00: 00: 00: 01),(OF|5@OF|00: 00: 00: 00: 00:
00: 00: 01->OF|1@OF|00: 00: 00: 00: 00: 00: 00: 03)],

OF|00: 00: 00: 00: 00: 00: 00: 01=[(OF|1@OF|00: 00: 00: 00: 00: 00: 00: 02-
>OF|4@OF|00: 00: 00: 00: 00: 00: 00: 01),(OF|1@OF|00: 00: 00: 00: 00: 00: 00:
03->OF|5@OF|00: 00: 00: 00: 00: 00: 00: 01),(OF|5@OF|00: 00: 00: 00: 00: 00:
00: 01->OF|1@OF|00: 00: 00: 00: 00: 00: 00: 03),(OF|4@OF|00: 00: 00: 00: 00:
00: 00: 01->OF|1@OF|00: 00: 00: 00: 00: 00: 00: 02)],

OF|00: 00: 00: 00: 00: 00: 00: 04=[(OF|2@OF|00: 00: 00: 00: 00: 00: 00: 04-
>OF|5@OF|00: 00: 00: 00: 00: 00: 00: 05),(OF|5@OF|00: 00: 00: 00: 00: 00: 00:
05->OF|2@OF|00: 00: 00: 00: 00: 00: 00: 04),(OF|1@OF|00: 00: 00: 00: 00: 00:
00: 04->OF|2@OF|00: 00: 00: 00: 00: 00: 00: 03),(OF|2@OF|00: 00: 00: 00: 00:
00: 00: 03->OF|1@OF|00: 00: 00: 00: 00: 00: 00: 04)],

OF|00: 00: 00: 00: 00: 00: 00: 05=[(OF|2@OF|00: 00: 00: 00: 00: 00: 00: 04-
>OF|5@OF|00: 00: 00: 00: 00: 00: 00: 05),(OF|5@OF|00: 00: 00: 00: 00: 00: 00:
05->OF|2@OF|00: 00: 00: 00: 00: 00: 00: 04),(OF|4@OF|00: 00: 00: 00: 00: 00:
00: 05->OF|2@OF|00: 00: 00: 00: 00: 00: 00: 02),(OF|2@OF|00: 00: 00: 00: 00:
00: 00: 02->OF|4@OF|00: 00: 00: 00: 00: 00: 00: 05)]
}
```

Get topology information with their property:

```
topologyManager.getEdges()

{
(OF|2@OF|00: 00: 00: 00: 00: 00: 00: 04->OF|5@OF|00: 00: 00: 00: 00: 00: 00:
05)=[State[1],Config[1],TimeStamp[creation:1422214094910],Name[s5-
eth5],BandWidth[10Gbps]],

(OF|1@OF|00: 00: 00: 00: 00: 00: 00: 02->OF|4@OF|00: 00: 00: 00: 00: 00: 00:
01)=[State[1],Config[1],Name[s1-eth4],BandWidth[10Gbps],TimeStamp[creation:
1422214094920]],

(OF|5@OF|00: 00: 00: 00: 00: 00: 00: 05->OF|2@OF|00: 00: 00: 00: 00: 00: 00:
04)=[State[1],Config[1],BandWidth[10Gbps],TimeStamp[creation:1422214094918],N
ame[s4-eth2]],
```

```
(OF|4@OF|00: 00: 00: 00: 00: 00: 00: 05->OF|2@OF|00: 00: 00: 00: 00: 00: 00:
02)=[State[1],TimeStamp[creation: 1422214094914],Config[1],BandWidth[10Gbps],
Name[s2-eth2]],

(OF|1@OF|00: 00: 00: 00: 00: 00: 00: 04->OF|2@OF|00: 00: 00: 00: 00: 00: 00:
03)=[State[1],Name[s3-eth2],Config[1],BandWidth[10Gbps],TimeStamp[creation:
1422214094921]],
.
.
.
}
```

Get properties which belongs spesific port:

```
switchManager.getNodeConnectorProps(outPort);

{
name=Name[s2-eth1], state=State[1],
config=Config[1],bandwidth=BandWidth[10Gbps]
}
```

Get statistic information about switch and its ports:

```
statisticsManager.getNodeConnectorStatistics(node);

[NodeConnectorStats[portNumber=OF|3@OF|00: 00: 00: 00: 00: 00: 00: 01,
receivePackets=4,
transmitPackets=8,
receiveBytes=300,
transmitBytes=728,
receiveDrops=0,
transmitDrops=0,
receiveErrors=0,
transmitErrors=0,
receiveFrameError=0,
receiveOverRunError=0,
receiveCrcError=0,
collisionCount=0],
NodeConnectorStats[portNumber=OF|2@OF|00: 00: 00: 00: 00: 00: 00: 01,
receivePackets=4,
transmitPackets=8,
receiveBytes=300,
transmitBytes=728,
receiveDrops=0,
transmitDrops=0,
receiveErrors=0,
transmitErrors=0,
receiveFrameError=0,
receiveOverRunError=0,
receiveCrcError=0,
collisionCount=0],
NodeConnectorStats[portNumber=OF|5@OF|00: 00: 00: 00: 00: 00: 00: 01,
receivePackets=7,
transmitPackets=6,
receiveBytes=686,
transmitBytes=588,
receiveDrops=0,
transmitDrops=0,
```

```
receiveErrors=0,
transmitErrors=0,
receiveFrameError=0,
receiveOverRunError=0,
receiveCrcError=0,
collisionCount=0],
NodeConnectorStats[portNumber=OF|4@OF|00: 00: 00: 00: 00: 00: 00: 01,
receivePackets=11,
transmitPackets=12,
receiveBytes=1078,
transmitBytes=1176,
receiveDrops=0,
transmitDrops=0,
receiveErrors=0,
transmitErrors=0,
receiveFrameError=0,
receiveOverRunError=0,
receiveCrcError=0,
collisionCount=0],
NodeConnectorStats[portNumber=SW@OF|00: 00: 00: 00: 00: 00: 00: 01,
receivePackets=0,
transmitPackets=0,
receiveBytes=0,
transmitBytes=0,
receiveDrops=0,
transmitDrops=0,
receiveErrors=0,
transmitErrors=0,
receiveFrameError=0,
receiveOverRunError=0,
receiveCrcError=0,
collisionCount=0],
NodeConnectorStats[portNumber=OF|1@OF|00: 00: 00: 00: 00: 00: 00: 01,
receivePackets=1214,
transmitPackets=1217,
receiveBytes=117852,
transmitBytes=118202,
receiveDrops=0,
transmitDrops=0,
receiveErrors=0,
transmitErrors=0,
receiveFrameError=0,
receiveOverRunError=0,
receiveCrcError=0,
collisionCount=0]]
```

Get flow rule information and statistics from switch:

```
statisticsManager.getFlows(node)


[FlowOnNode[flow=Flow[match=Match[fields={
                                          DL_TYPE=DL_TYPE(2048),
                                          NW_DST=NW_DST(10.0.0.1,
                                          255.255.255.255)
},
matches=2176],
actions=[SET_DL_DST[address=ba2f89c4a9cb],
OUTPUT[OF|1@OF|00: 00: 00: 00: 00: 00: 00: 01]],
```

```
priority=1,
id=0,
idleTimeout=0,
hardTimeout=0],
tableId=0,
sec=425,
nsec=348000000,
pkt=5,
byte=490],
FlowOnNode[flow=Flow[match=Match[fields={
                                                DL_TYPE=DL_TYPE(2048),
                                                NW_DST=NW_DST(10.0.0.5,
                                                255.255.255.255)
},
matches=2176],
actions=[OUTPUT[OF|4@OF|00: 00: 00: 00: 00: 00: 00: 01]],
priority=1,
id=0,
idleTimeout=0,
hardTimeout=0],
tableId=0,
sec=424,
nsec=498000000,
pkt=5,
byte=490]]
```

Installing flow rule whit using controller interfaces to create service chain:

```
Match match = new Match();
List<Action> actions = new ArrayList<Action>();

match.setField(MatchType.DL_TYPE, EtherTypes.IPv4.shortValue());
match.setField(MatchType.NW_SRC, InetAddress.getByName(source.getIp()));
match.setField(MatchType.NW_DST, InetAddress.getByName(dest.getIp()));
match.setField(MatchType.NW_PROTO,IPProtocols.getProtocolNumberByte(dest.getP
rotocol()));
match.setField(MatchType.TP_SRC, source.getPort());
match.setField(MatchType.TP_DST, dest.getPort());

actions.add(new SetNwDst(InetAddress.getByName(destMachineIp)));
actions.add(new SetDlDst(destMachineMac));
actions.add(new Output(outport));
Flow flow = new Flow(match, actions);
flow.setIdleTimeout((short) 5);
flow.setHardTimeout((short) 0);
flow.setPriority(LB_IPSWITCH_PRIORITY);

String policyName = source.getIp() + ":" + source.getProtocol() + ":" +
source.getPort();

String flowName = "[" + policyName + ":" + source.getIp() + ":" +
dest.getIp() + "]";


FlowEntry fEntry = new FlowEntry(policyName, flowName, flow, sourceSwitch);

ruleManager.installFlowEntry(fEntry);
```

4.1.4 SDN-controlled device-to-device communications

4.1.4.1 Updates on architecture mapping

Figure 36 presents the additional components to add in the SIGMONA reference architecture, required by the SDN-controlled IP wireless mesh network. Regarding the user plane (data plane), we have in addition to the eNB a WiFi access point, offering the possibility to offload eNB. On the other hand, for the control plane, multiple services have to be added. In the SDN controller, we must implement both congestion detection service and application mesh client service. These latter are managed by the Mesh Coordinator Server placed on an upper layer (Network Services). Communications between client-server applications are ensured through APIs.

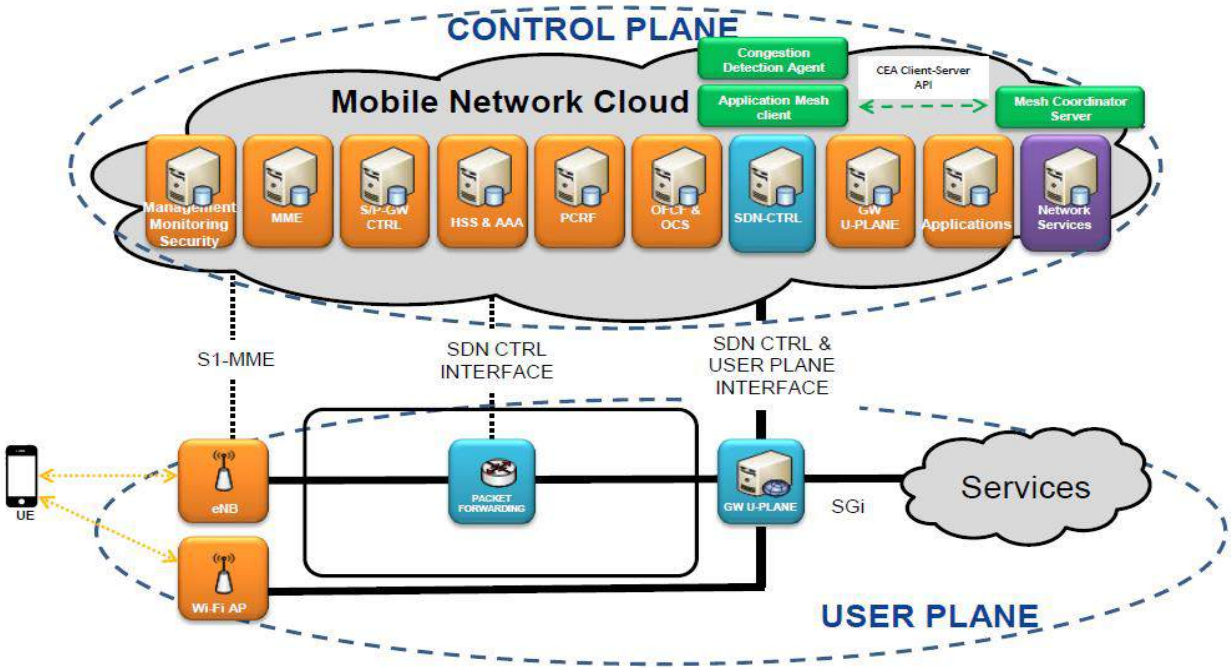


Figure 36 – SDN-controlled IP WMN (SIGMONA reference architecture)

A complete SDN-controlled IP WMN use case requires the following components in an SDN-layered reference architecture model, illustrated in Figure 37:

The main required functions in the SDN architecture layers are the following:

1. Infrastructure or datapath-layer: UEs SDN-enebaled, eNB, WiFi Access point SDN
2. Controller-layer: congestion detection, performance estimation, application mesh client
3. SDN Application-layer: Mesh Coordinator Server
4. Management-layer: CEA Mesh Service Manager

These functions perform the following tasks:

• 1-2: Optimal routing, congestion monitoring, performance estimation (Cellular/Mesh networks)

• 2-3: Coordinate and manage controller layer service according to the general structure described in 3.2.4.2

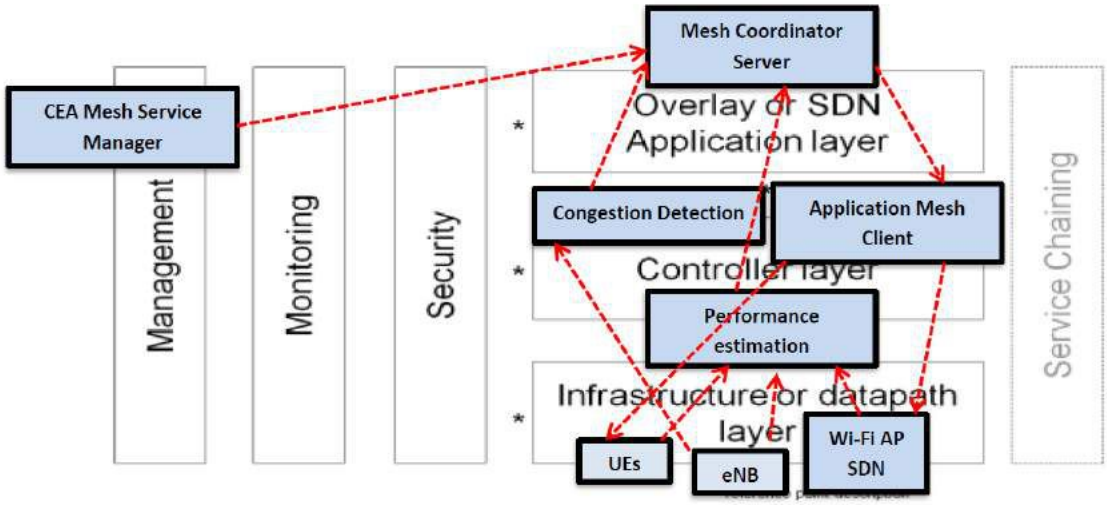


Figure 37 - SDN-controlled IP WMN (SDN-layered reference model view).

• 3-4: Resource status (monitoring) and inventory management unit to create chains

Mapping to the ETSI NFV architectural framework:

Figure 38 includes the mapping of our research contribution into SDMN architecture. This mapping results in multiple components that utilize SDN/NFV as basic technology. The goal of such proposition is to ensure future proof solution taking into standardization guideline considerations such as the ETSI ISG NFV architectural framework.

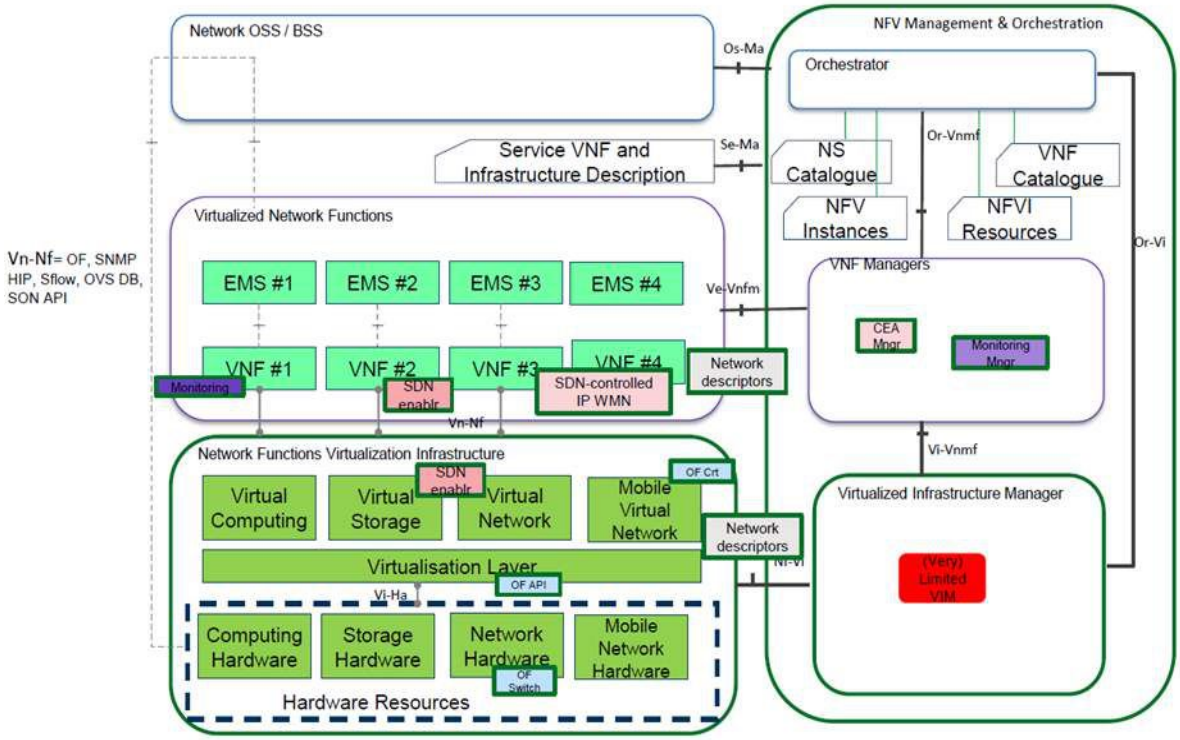


Figure 38 - SDN-controlled IP WMN (ETSI NFV architectural framework)

Three communication channels have been identified:

- The northbound interface to provide communication to VNFm, orchestration services and Hypervisor network components,
- The East/west interface to provide communication between SDN controllers,
- The Southbound interface (OpenFlow) to communicate to SDN switches.

Data Offloading SDN-controlled IP wireless mesh: Through this NFV, we address the opportunity of offloading the Radio Access Network by the use of IP Wireless Mesh Network (e.g. Wi-Fi). Here we consider SDN control of IP communications in the edge networks, meaning that smartphones are SDN capable. It is commonly assumed that such a target would be handled by end-terminals themselves as a completely distributed system without mobile network supervision. This research topic investigates whether the mobile network operator can keep control of these communications to redirect traffic to a different access network (e.g., fixed).

Note: Much more details and explanations can be found in the dedicated document (D1.2 from the WP1).

4.1.4.2 Detailed interface descriptions

The interface (southbound), in addition to the classical informations needed for optimal routing, flows monitoring, resources management and so on, must capture the following specific informations:

- Bandwidth remaining percent for each of the LTE network elements, still within the context of congestion monitoring.
- An information allowing to distinguish between UEs, WiFi AP, LTE forwarding devices. Indeed, this information is paramount for the SDN controller in order to achieve the following tasks:
    - Dedicated processing in function of the network element type.
    - Detect a potential mesh networks based on the UEs.
    - Redirect traffic to WiFi APs for the case of offloading.
- Detailed informations of the WiFi interfaces present in the mesh network, allowing to estimate the performance of the mesh network.
- The Channel Quality Indicator (CQI) delivered by the eNB. As the name implies, it is an indicator carrying the information on how good/bad the communication channel quality is. It will allow deducing the performance of the eNB.
- LTE Buffer Status Report (BSR), allowing to anticipate the traffic load on in the cellular network.

4.2 Microscopic Traffic Management

4.2.1 DiffServ QoS architecture in SDMNs

4.2.1.1 Updates on architecture mapping

Figure 39 illustrates the additional components in the SIGMONA reference architecture figure, required by dynamic QoS control. Queueing discipline (Qdisc) configurations are needed in the datapath in each SDN switch, in order to set up the per hop behavior for different traffic classes. In the mobile network cloud, communication between the PCRF and SDN controller is required for PCRF-based QoS service configuration and authorization.

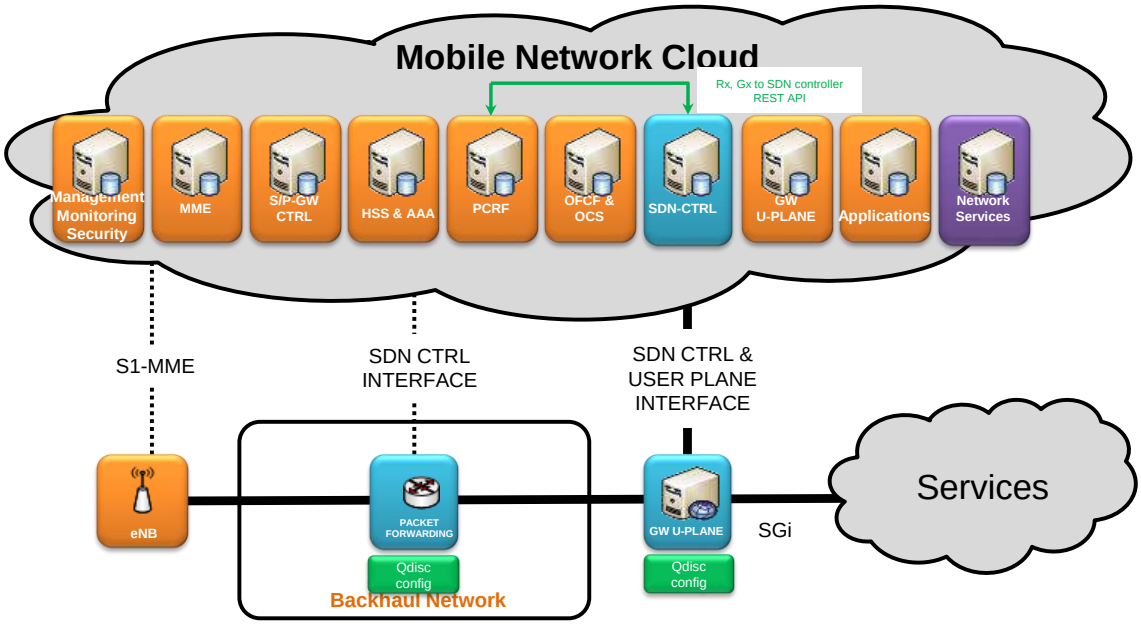


Figure 39 – DiffServ QoS in SIGMONA reference architecture.

A complete DiffServ QoS use case requires the following components in an SDN-layered reference architecture model, illustrated in Figure 40:

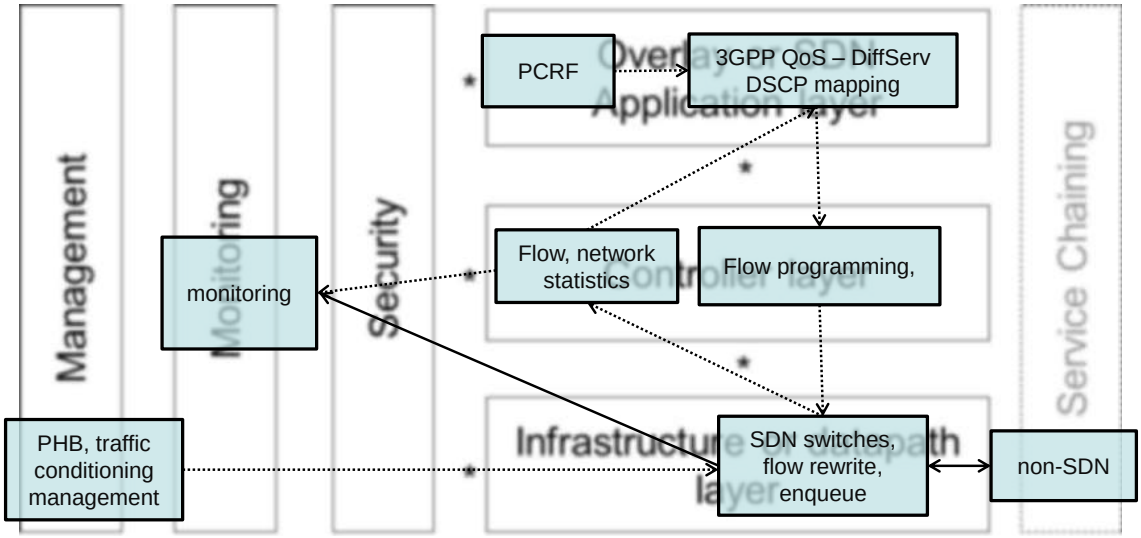


Figure 40 – DiffServ QoS provision using PCRF (SDN-layered reference model view).

The main required functions in the SDN architecture layers are the following:

1. Infrastructure or datapath-layer: OpenFlow switches, DSCP or other marking, packet scheduling
2. Controller-layer: flow programming, flow statistics
3. SDN Application-layer: PCRF–based QoS authorization and control for different traffic classes
4. Management-layer: DiffServ per-hop behavior configuration in switches, traffic conditioning
5. Monitoring: monitoring flow statistics, QoS/QoE measurements

Required reference points are the following:

Ref. point ‘1-2’: OpenFlow protocol, e.g., FlowMod, PacketIn

Ref. point ‘2-3’: translation between Gx (and Rx) interfaces and SDN controller REST APIs

Ref. point ‘3-4’: Qdisc configuration in SDN switches through management-layer.

Ref. point ‘1-5’: QoS/QoE monitoring

Ref. point ‘2-5’: flow, network statistics from controller

4.2.1.2 Detailed interface descriptions

Appendix A.2.4 of IR3.4b [22] contains a detailed description of the unsolicited PCC rule provisioning over the Gx interface between the L3 Routing and DiffServ control application and the PCRF. The following link contains the description of the Diameter Base protrocol and Gx application commands.

4.3 Integration of Technologies

The ETSI NFV Industry Specification Group is on track to define an NFV ecosystem using the works of different standard organizations and identifying new gaps for them. The ETSI NFV reference architecture [21] provides a good reference for the SDMN architecture. NFV architecture has three main parts. (1) Network Function Virtualization Infrastructure (NFVI) contains the compute (CD), hypervisor (HD) and infrastructure network domains (IND), which realize host and network virtualization above physical resources. (2) Virtual network Functions (VNF) are made of components, which are attached to container interfaces of virtual machines (VM) and are connected by virtual networks (VN). (3) Management and Orchestration (MANO) of NFVI and VNF are realized through Virtual Infrastructure Manager (VIM) and VNF Manager (VNFM), respectively. The Orchestrator is the high-level hook for management, it is aware of NFVI resources and VNF instances. Further details on the NFV reference architecture can be found in ETSI NFV Open Area [21].

Figure 41 maps our Software Defined Mobile Networking (SDMN) components to the NFV architecture. It is mainly a functional diagram, but also illustrates that different software-based functions of SDMN can run as VNFs. Resource, traffic and mobility management require changes in and between IND, VIM NFVI Network control, VNFM and Orchestrator.

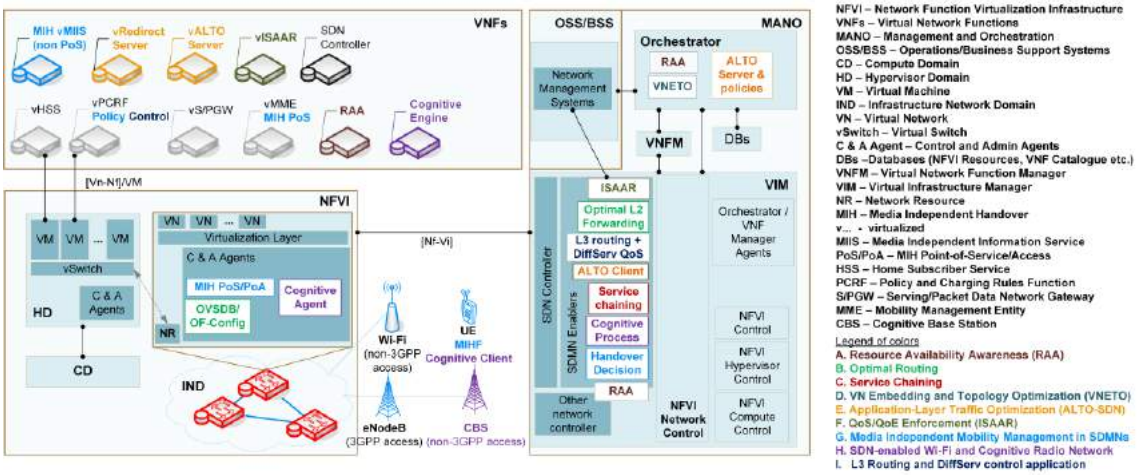


Figure 41 – High-level architecture

The network infrastructure of the operator contains parts outside the NFVI, which cannot be virtualized, e.g., remote radio heads of base stations or the physical layer of Wi-Fi access points. On the other hand, the base band unit, software part, controls and admin (C&A) agents of network components are considered as parts of the IND.

In addition to the three main functional blocks of the NFV architecture, NFV architecture must be compatible with service providers' existing management systems, Operational and Business Support Systems (OSS/BSS) and their legacy network equipment in order to support management functions such as network inventory management, service provisioning, network configuration and fault management. The SDN and other network controllers provide abstractions and Northbound Application Programming Interfaces (APIs) for the NFVI network control of the functions required for management and control of network resources.

The SIGMONA project proposes multiple resource, traffic and mobility management applications, most of them requiring new SDN applications, called SDMN enablers in the figure.

Traffic management solutions proposed by BME-MIK are the following:

E. Application-Layer Traffic Optimization in SDNs (in orange)

- ALTO server: it can be deployed as a VNF. From functional aspect, it could be part of ETSI NFV MANO, supporting the endpoint selection for any type of distributed service
- HTTP redirect server: it can be deployed as a VNF. Further enhancement could be that HTTP redirect is service running in certain SDN switches.
- ALTO client: SDMN enabler application running on top of the NB interface of SDN controllers

I. L3 Routing and DiffServ QoS (in dark blue)

- PCRF running as a VNF realizing the server side of Diameter 3GPP Gx application
- Routing and DiffServ control application: SDMN enabler application, realizing the client side of Diameter 3GPP Gx application. It may push static routes, OF meter settings in the SDN switches through the controller, and Queueing discipline and traffic filter settings using remote configuration protocols of the switches.

The proposed concepts have been mapped to the ETSI NFV framework. We can see that certain components could utilize the service of other components. Optimized Routing, e.g., could be used by any component for packet forwarding in SDN network segments. ALTO-SDN could serve Service Chaining if the services on the forwarding path were distributed. If ALTO network and cost maps covered different types of access networks, ALTO-SDN service could be used to guide access network selection of UEs. Furthermore, combined virtual mobile core network function placement and topology optimization probably will have influence on possible service endpoints, VNFs, virtual network topologies, influencing the VN Forwarding paths, ALTO network and cost maps. MIH and/or Cognitive Engine driven proactive handovers should serve two purposes, i.e., HO due to loss of network, or smart traffic steering considering the policies of the operator and the subscribers. The dependencies of the different components hence should be further analyzed in our future work including the assessment of the possible gains of joint operations, and integration using appropriate interfaces.

5. Conclusions

In this deliverable, we have examined technologies and architectures for traffic and resource management in software defined mobile networks.

The main research challenges have been analyzed and related research directions have been defined resource and traffic management topics. The solutions for resource management have been presented by examining resource availability in SDMN by collecting information regarding virtual and physical networks for monitoring status. Next, we have proposed architectures, technologies and solutions for macroscopic traffic management based on energy efficient routing algorithm which reduces the number of active switches depending on the traffic, application-layer traffic optimization (ALTO) for the orchestration of endpoint selection for distributed services, joint traffic and resource management for service chaining that enable management of flows based on congestion information about network in terms of traffic management, SDN-controlled device-to-device communications based on congestion monitoring approach. For microscopic traffic management, we have analyzed the feasibility dynamic DiffServ-based QoS control using legacy PCRF, QoE/QoS enforcement in SDMN, the use of SDN to provide a guaranteed QoS to premium users in a VoIP system congested by both from VoIP and non-VoIP background traffic.

In the last part of the deliverable, we have defined architectural modifications required by all proposed solutions individually for SDN-layered architecture model and then we have proposed an integrated model including proposed traffic and resource management solutions for SDN and NFV based mobile network architecture.

[1] R. Alimi (Ed.), R. Penno, Y. Yang, "*ALTO Protocol"*, IETF RFC 7285, September, 2014.

[2] Knoll T. M., Eckert M.: ISAAR (Internet Service Quality Assessment and Automatic Reaction) a QoE Monitoring and Enforcement Framework for Internet Services in Mobile Networks; *4th International Conference, MONAMI 2012*;

[3] Tao Feng, Jun Bi, Ke Wang, Joint Allocation and Scheduling of Network Resource for Multiple Control Applications in SDN, IEEE ICC, 2014.

[4] H.E. Egilmez and A.M. Tekalp, ''Distributed QoS Architectures for Multimedia Streaming over Software Defined Networks", *IEEE Trans. on Multimedia*, Accepted with Minor Revision.

[5] Junjie Zhang, Kang Xi , Min Luo, H. Jonathan Chao, "Load Balancing for Multiple Traffic Matrices Using SDN Hybrid Routing", *IEEE 15th International Conference on High Performance Switching and Routing,* 2014.

[6] Z. Faigl (Ed.), "State-of-the-art and challenges of traffic, resource and mobility management in software-defined mobile networks", SIGMONA Deliverable D3.1, November, 2014.

[7] V. Gurbani, M. Scharf, T. V. Lakshman, V. Hilt and E. Marocco "Abstracting network state in Software Defined Networks (SDN) for rendezvous services", *in Proceedings of the IEEE International Conference on Communications (ICC),* 2012, pp. 6627–6632.

[8] J. Medved et al., „ALTO Network-Server and Server-Server APIs", http://tools.ietf.org/id/draft-medved-alto-svr-apis-00.txt

[9] P. Racz, Z. Despotovic, „An ALTO Service based on BGP Routing Information", http://tools.ietf.org/id/draft-racz-bgp-based-alto-service-00.txt

[10] H. Gredler, J. Medved, S. Previdi, A. Farrel, S. Ray, „North-Bound Distribution of Link-State and TE Information using BGP", http://tools.ietf.org/html/draft-ietf-idr-ls-distribution-04

[11] Y. Lee, G. Bernstein, Grotto Networking, D. Dhody, T. Choi, "ALTO Extensions for Collecting Data Center Resource Information", http://tools.ietf.org/id/draft-lee-alto-ext-dc-resource-03.txt

[12] Gautam Khetrapal, Saurabh Kumar Sharma, "Demystifying routing services in software-defined networking," Aricent, Whitepaper, 2013.

[13] Z. Faigl, Zs. Szabó, R. Schulcz, "Application-layer traffic optimization in software-defined mobile networks: a proof-of-concept implementation", *16th International Telecommunications Network Strategy and Planning Symposium* (Networks 2014), Madeira Island, Portugal, Sep. 17-19, 2014.

[14] W. Tan, J. Zhang, C. Peng, B. Xia, and Y. Kou, "Sdn-enabled converged networks," Wireless Communications, IEEE, vol. 21, no. 6, pp. 79–85, December 2014.

[15] W. Su, C. Liu, C. Lagoa, H. Che, K. Xu, and Y. Cui, "Integrated, distributed traffic control in multidomain networks," Control Systems Technology, IEEE Transactions on, no. 99, pp. 1–1, 2014.

[16] B. Nunes, M. Santos, B. de Oliveira, C. Margi, K. Obraczka, and T. Turletti, "Software-defined-networking-enabled capacity sharing in user-centric networks," Communications Magazine, IEEE, vol. 52, no. 9, pp. 28–36, September 2014.

[17] F. Paganelli, M. Ulema, and B. Martini, "Context-aware service composition and delivery in NGSONs over SDN," Communications Magazine, IEEE, vol. 52, no. 8, pp. 97–105, Aug 2014.

[18] "Ieee standard for the functional architecture of next generation service overlay networks," IEEE Std 1903-2011, pp. 1–147, Oct 2011.

[19] W. Wendong, Q. Qinglei, G. Xiangyang, H. Yannan, and Q. Xirong, "Autonomic qos management mechanism in software defined network," Communications, China, vol. 11, no. 7, pp. 13–23, July 2014.

[20] General information about the ETSI ISG NFV: http://www.etsi.org/technologies-clusters/technologies/nfv

[21] ETSI ISG NFV open area for sharing the draft documents: http://docbox.etsi.org/ISG/NFV/Open/Latest_Drafts/

[22] Z. Faigl (ed.), "Validation and demonstration plan", SIGMONA Internal report IR3.4b, May, 2015.

[23] J. L. Deng. Introduction to grey system theory. J. Grey Syst., 1(1):1–24, November 1989.

[24] A. Huszak and S. Imre. Eliminating Rank Reversal Phenomenon in GRA-Based Network Selection Method. In Proceedings of IEEE International Conference on Communications (ICC), 2010, pages 1–6, Cape Town, South Africa, 23-27May 2010.

[25] F.A. Lootsma. Multi-criteria decision analysis via ratio and difference judgement, volume 29 of Applied Optimization. Kluwer Academic, Dordrecht, 1999.

[26] E. Triantaphyllou and K. Baig. The impact of aggregating benefit and cost criteria in four MCDA methods. IEEE Transactions on Engineering Management, 52(2):213–226, May 2005.

[27] Phuoc Nguyen Tran and Nadia Boukhatem. The distance to the ideal alternative (DiA) algorithm for interface selection in heterogeneous wireless networks. In Proc. of the 6th ACM Int. Symp. on Mobility management and wireless access 2008 MobiWac '08, pages 61–68, Oct. 2008.

[28] R.Venkata Rao. Improved multiple attribute decision making methods. In Decision Making in Manufacturing Environment Using Graph Theory and Fuzzy Multiple Attribute Decision Making Methods, Springer Series in Advanced Manufacturing, pages 7–39. Springer London, 2013.

[29] T. L. Saaty. The Analytic Hierarchy Process: Planning, Priority Setting, Resource Allocation. McGraw-Hill, New York, St. Louis, San Francisco, 1980.

[30] Ying-Ming Wang and Ying Luo. On rank reversal in decision analysis. Mathematical and Computer Modelling, 49(5–6):1221–1229, 2009.

[31] Jonathan Barzilai and Boaz Golany. AHP rank reversal, normalization and aggregation rules. Infor-Information Systems and Operational Research, 32(2):57–64, 1994.

[32] P. TalebiFard and V.C.M. Leung. A dynamic context-aware access network selection for handover in heterogeneous network environments. In Proc. of the 2011 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pages 385 –390, April 2011.

[33] Q. Song and A. Jamalipour. A network selection mechanism for next eneration networks. In Proc. of the 2005 IEEE International Conference on Communications (ICC 2005), volume 2, pages 1418–1422 Vol. 2, may 2005.

[34] F. Bari and V. Leung. Multi-Attribute Network Selection by Iterative TOPSIS for Heterogeneous Wireless Access. In Proc of the 4th IEEE Consumer Communications and Networking Conference, 2007 (CCNC 2007), pages 808–812, January 2007.

[35] C. T. R. Hager. Context Aware and Adaptive Security for Wireless Networks. PhD thesis, Virginia Polytechnic Institute and State University, November 2004.

[36] H. Johnson. Toward Adjustable Lightweight Authentication for Network Access Control. PhD thesis, Blekinge Institute of Technology, Ronneby, Sweden, December 2005.

[37] B. S. Ghahfarokhi and N. Movahhedinia. Context-Aware Handover Decision in an Enhanced Media Independent Handover Framework. Wireless Personal Communications, 68:1633–1671, February 2013.

[38] K. Bala M. K. and B.R. Tamma. An enhanced media independent handover framework for heterogeneous wireless networks. In Proc. of the 12th International Conference on Intelligent Systems Design and Applications (ISDA), 2012, pages 610–615, November 2012.

[39] 3GPP, "Policy and charging control architecture (Release 12)", TS 23.203, v12.2.0, September 2013.

[40] "Network Congestion Management: Considerations and Techniques", Sandvine Industry Whitepaper. [Online]. https://www.sandvine.com/downloads/general/whitepapers/network-congestion-management.pdf

[41] A. Maeder, S. Schmid, Z. Yousaf, "Towards User-Plane Congestion Management in LTE EPS", [Online]. Available at: http://www.ikr.uni-stuttgart.de/Content/itg/fg524/Meetings/2012-03-13-Muenchen/02_ITG524_Munich_Maeder.pdf

[42] A. Kuzmanovic, A. Mondal, S. Floyd, K. Ramakrishnan, RFC 5562 – "Adding Explicit Congestion Notification Capability to TCP's SYN/ACK Packets". June 2009