Enabling Network Applications for Multi-Service Programmability in a Disaggregated RAN

Chia-Yu Chang and Navid Nikaein
Communication Systems Department, EURECOM, France
Email: firstname.lastname@eurecom.fr

Abstract

Network slicing is one of the key enablers in providing the required flexibility for realizing the service-oriented 5G vision and achieving the desired levels of isolation and sharing spanning multiple subnets, i.e., core network, transport network and radio access network (RAN). In this article, we highlight the design elements of a proposed RAN runtime slicing system to enable flexible slice customization on top of disaggregated RAN infrastructures. Moreover, a runtime software development kit (SDK) is introduced to facilitate the development of agile control applications able to monitor, control, and program the underlying RAN modules. Also, we highlight the chaining of single- and cross-domain control applications to form the application plane and implement sophisticated control logics. Finally, a prototype of the proposed RAN runtime and the runtime SDK is provided based on the OpenAirInterface and Mosaic5G platforms to demonstrate how slicing and programmability can be achieved in two use cases.

Index Terms

RAN slicing, 5G, Multi-service chaining, Control application, SDK, API.

I. Introduction

To realize the *service-oriented* vision toward fifth generation (5G), the evolving network slicing paradigm can be adopted to create self-contained logical networks on top of shared physical and virtualized infrastructures. According to the third generation partnership project (3GPP) TR28.801, each network slice instance includes a set of network functions and resources spanning multiple subnets, i.e., core network (CN) and radio access network (RAN), which are arranged and configured to form a complete logical

network that meets certain service level agreements (SLAs). A crucial aspect of flexibly customizing each slice and satisfying the end-to-end (E2E) service requirements is related to the RAN domain, i.e., *RAN slicing*, delivering the RAN *as-a-service* on a per-use- case basis.

To control and manage multiple slice instances at the RAN domain, a *multi-service execution environment* is required to provide the customized virtual views on top of the same physical RAN infrastructure, which is tailored to per-slice requirements. Such a flexible customization relies on two principles: *softwarization* and *virtualization*, which constitute the foundations for a multi-service and multi-tenant architecture. Softwarization decouples software from hardware and control-plane (CP) from user-plane (UP) processing, while virtualization facilitates the instantiation of several customized network functions over a common infrastructure.

Note that the underlying RAN infrastructure can be either a monolithic base station (BS) or *disaggregated*, where it is decomposed into radio unit (RU) equipment, distributed unit (DU) and centralized unit (CU), with the functional splits in-between, as with the split options defined by the 3GPP in TR38.801. Such disaggregation not only retains the benefits of coordinated and cooperative processing at the cloud infrastructures, but also extends resource abstraction schemes among multiple RAN nodes and the interconnected transport networks to compose the RAN. Additionally, slice-customized functions including CP and UP processing and control logic (CL) can be chained among disaggregated RAN entities to reflect the slice-specific requirements, with the controlled access to a portion of resources and CP/UP states in a virtualized form.

Several standardization bodies, such as the 3GPP, highlight the E2E network slicing notion to fulfill the 5G service-oriented vision. Particularly, the 3GPP addressed network slicing from the architecture perspective in TS23.501 and the corresponding RAN slicing in TR38.801. Moreover, the RAN slicing fulfills several 5G RAN design requirements in [1] through the software-defined RAN (SD-RAN) architecture. Stemming from the software defined networking (SDN) concept, the SD-RAN aims to decouple the CP and UP processing and is exploited by the FlexRAN platform [2] utilizing the customized south-bound application platform interface (API) for the flexibly programmable CP processing with different levels of centralization. Several RAN slicing studies have been conducted. The BS hypervisor provided in [3] isolates the slice-specific CL and shares radio resources among different slices. Additionally, the RAN runtime presented in [4] can customize and multiplex in aspects of resource, state and processing over a disaggregated RAN. Even with the aforementioned RAN slicing prototypes, the focus today is still on ways to virtualize radio resource, isolate slice performance, and orchestrate the service. This work extends the efforts of current studies and aims to provide the required software development kit (SDK) primitives over the sliced RANs to enable flexible, customized, and plug-and-play (P&P) CL programmability.

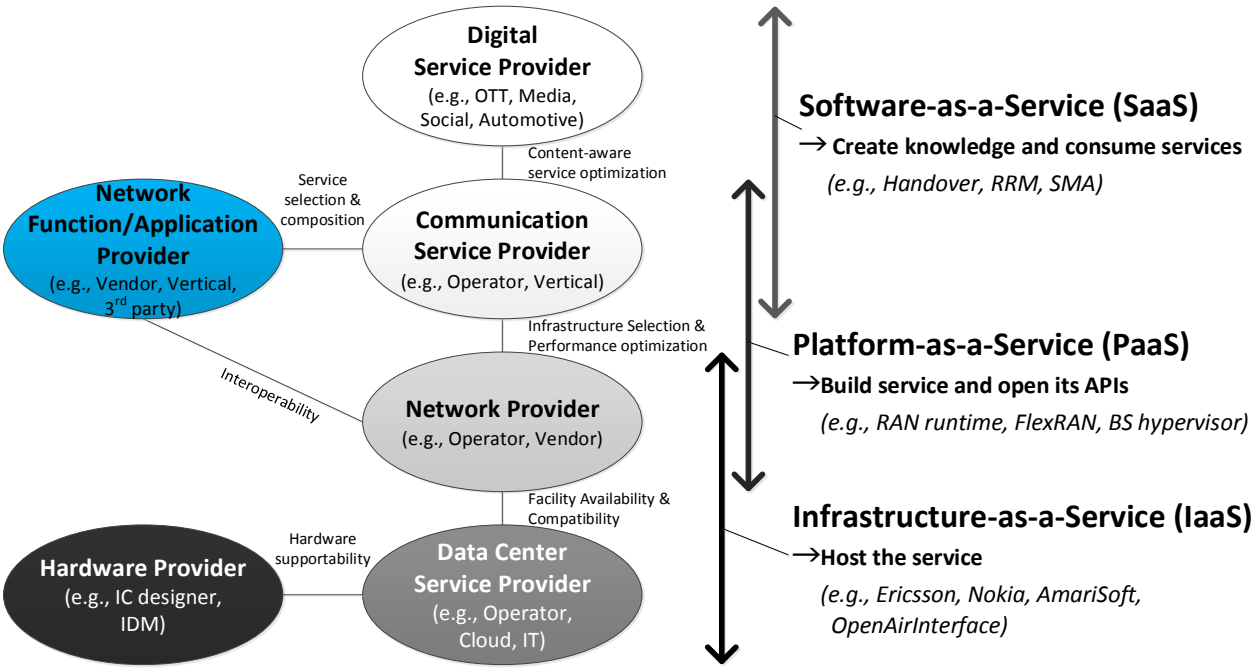


Fig. 1: The three as-a-service levels among different providers in the value chain.

To highlight the importance of CL programmability, the relationship between different providers in the value chain transformation of the telecommunication industry is illustrated in Fig. 1, which is aligned with the 3GPP proposal in TR28.801. The providers of network infrastructure, network function and application, and communication and digital service are decoupled to allow a cost-effective and flexible network composition. Three overlapping levels are observed. The Infrastructure-as-a-Service (IaaS) provides the programmable physical and/or virtual infrastructures (e.g., software-defined radio and x86-based infrastructure) to host the RAN services, ranging from commercial to open source. The Platform-as-a-Service (PaaS) extends IaaS in support of monitoring, control, orchestration, and network function virtualization and provides open APIs and the slice-friendly development environment. The afore-mentioned FlexRAN, BS hypervisor and RAN runtime belong to this category. The Software-as-a-Service (SaaS) consumes the programmable control applications, such as the radio resource management (RRM) and spectrum management application (SMA), to provide the CLs. For instance, the programmability of spectrum management and RRM can allocate available spectrum and specific radio resources to deliver the mobile broadband over-the-top (OTT) digital service with certain quality of experience (QoE) levels. To deploy P&P control applications for specific use cases, the underlying platform package and SDKs are emphasized by several works. The network store concept introduced in [5] features the developed network functions and applications for each slice utilizing the provided SDK. In [6], a Python-based SDK

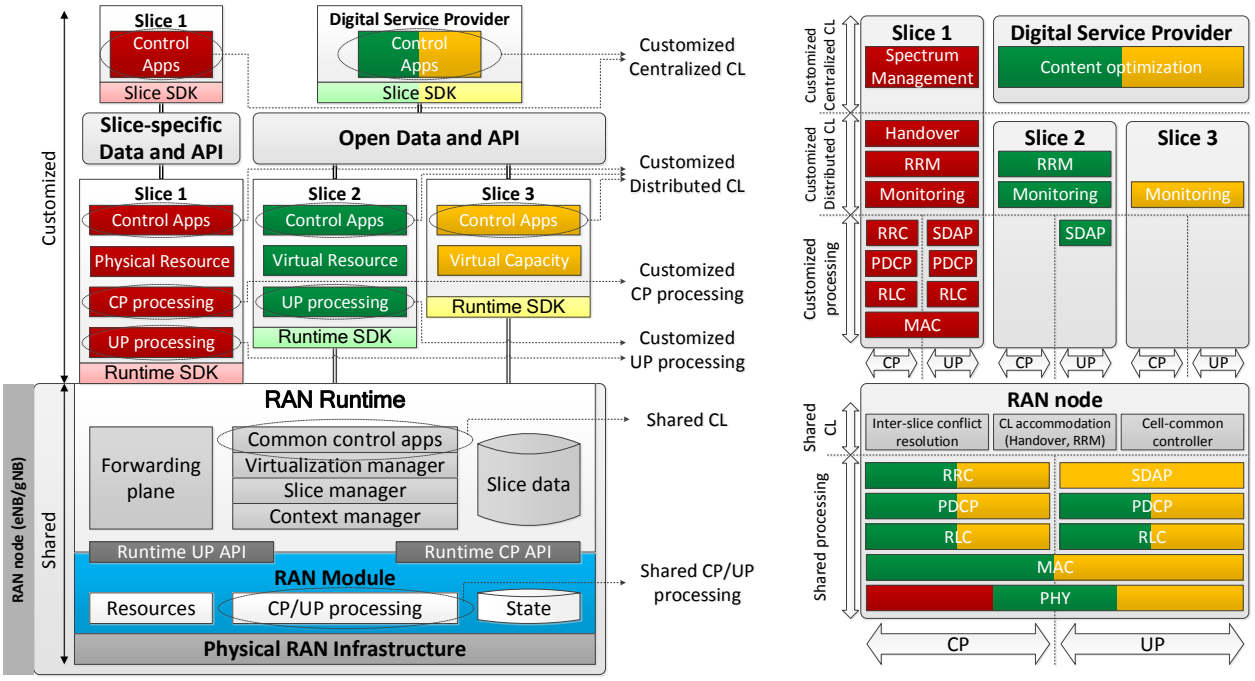


Fig. 2: The architecture of the RAN runtime slicing system (left) and an example with three slice instances (right).

empowers network applications development exploiting several programming primitives through REST or Python API.

In summary, this article makes the following contributions:

- Review the RAN runtime slicing system and introduce how the customized CLs are enabled by common/dedicated control applications and the SDK (Section II);
- Provide the control application execution environment leveraging the SDK and application plane for flexible and sophisticated CLs. (Section III);
- Build several control applications on top of OpenAirInterface and Mosaic5G platforms and present the potentials to chain applications in two use cases (Section IV).

II. RAN RUNTIME SLICING SYSTEM

We hereby introduce how the RAN runtime provides a multi-service execution environment targeting flexible customization and sharing at the RAN node, i.e., long-term evolution (LTE) evolved NodeB (eNB) or new radio next generation NodeB (gNB). Then, each running slice can develop and plug its control applications on top of the runtime SDK, interact with the RAN modules to access its resource/state, and control the underlying behaviors, as shown in the left part of Fig. 2.

A. RAN runtime

First, the network slice templates [7] can be used to describe the business applications, including the specific use case (e.g., public safety), deployed topology (e.g., geographical region), corresponding SLA, policies (e.g., resource isolation) and requirements (e.g., E2E latency). Such template can be further translated into the network slice description (NSD) defined by the European Telecommunications Standards Institute (ETSI), including the related physical/virtual network functions (PNFs/VNFs) with their dependencies, monitoring parameters, key performance indicators (KPIs), and deployment attribute (e.g., life-cycle events). These NSDs can be utilized to orchestrate and manage the network services at different technology domains, e.g., the RAN domain and CN domain.

The RAN runtime can allow slice owners to (a) orchestrate and manage their slices, (b) perform customized CLs (e.g., handover decisions) and/or CP/UP processing, (c) operate on a set of virtual resources (e.g., resource block), capacity (e.g., data rate) or latency (e.g., queuing delay), and (d) access their CP/UP state (e.g., user identity) revealed by the RAN runtime. Such a slice owner corresponds to the communication service provider illustrated in Fig. 1 that consumes the network provided by the operators. The RAN runtime also enables the operators to manage the underlying RAN module, enforce slice-specific policies, perform access control, and provide the BS-common (i.e., slice-independent) services to users. Note that each RAN module comprises a subset of RAN functions with the associated resources and states to perform a portion of the RAN processing over the common and/or specialized physical RAN infrastructure (e.g., hardware accelerator).

Within the RAN runtime, four services are provided over the **slice data**: (1) **slice manager**, (2) **virtualization manager**, (3) **context manager**, and (4) **common control applications**. Slice data include both slice context (e.g., basic information to instantiate a slice service, such as its identity, user context and slice association information) and the RAN context (e.g., CP/UP state information and module primitives), which are used to customize and manage a slice in terms of the required resources, states, processing, and users. Such slice data can be transferred or shared among different RAN runtime instances because of the user and network dynamics, such as mobility and service updates [8].

Based on the slice context, the slice manager determines the CP/UP processing chain for each slice and each traffic flow, and programs the **forwarding plane** to direct the input and output data paths across the shared processing operated by the underlying RAN module and the customized processing performed by each slice. An example with three slice instances is depicted in the right part of Fig. 2. For slice 1, both the CP and UP processing are separated into customized processing (radio resource control [RRC], service data adaptation protocol [SDAP], packet data convergence protocol [PDCP], radio link control

[RLC], medium access control [MAC] layers) and shared processing (Physical [PHY] layers), while slice 2 only customizes its SDAP functions for UP processing. By contrast, slice 3 relies on the shared CP/UP processing without any customization. Additionally, the resulted RAN service is operated by the slice manager in support of service continuity. For example, a slice that performs the customized UP processing can opt in for a shared processing to reduce its operational expense, which causes changes in the chained PNFs/VNFs. Finally, the slice manager is responsible for taking actions based on predefined policy rules when any conflict occurs at the slice or user level.

The virtualization manager provides the required level of isolation and sharing among slices. Specifically, it partitions resources and states, maps physical resources and states to and from the virtualized ones, and reveals virtual resources and states to a slice, which are decoupled from the physical ones. Hence, the slice-specific control application can execute its customized CLs over its virtualized network view. Take the three different radio resource abstraction types in Fig. 2 as examples. Slice 1 requests the physical resource blocks without any abstraction; slice 2 gets the virtual resource blocks that may not be mapped one-to-one toward the physical ones; and slice 3 receives the virtualized capacity (e.g., the aggregated throughput).

The context manager performs the CRUD operations (i.e., create, read, update, and delete) on both the slice and RAN contexts. To create a slice context, it first performs slice admission control based on its required processing, resources, and states. Upon admission control, the RAN context is used by the context manager to (a) register slice-specific life-cycle primitives to the slice manager, and (b) make a request of resources and/or performance to the virtualization manager. Afterwards, a slice can start to consume the services provided by the RAN runtime.

The common control applications provide a shared CL for multiple slices. It can accommodate the customized control decisions from different slice-specific control applications, resolve their conflicts, and enforce a feasible policy for the underlying RAN node. For instance, the two customized RRM applications of slices 1 and 2 in the right part in Fig. 2 will leverage the inter-slice conflict resolution and CL accommodation to provide their specific CLs through the cell-common controller. Hence, the customized CLs of each slice can be applied, which will be further elaborated in Section III.

Based on the aforementioned services provided by the RAN runtime, customized slice service can be initiated leveraging the exposed runtime SDK and runtime CP/UP API (cf. Fig. 2). The former is in the north-bound toward each instantiated slice, and each slice is connected to the RAN runtime through the communication channel as a separated process, whether it is local or remote. This allows each slice to span its own execution environment, either at the host or guest level, leveraging operating system and virtualization technologies, such as container or virtual machine. The latter is in the south-bound toward

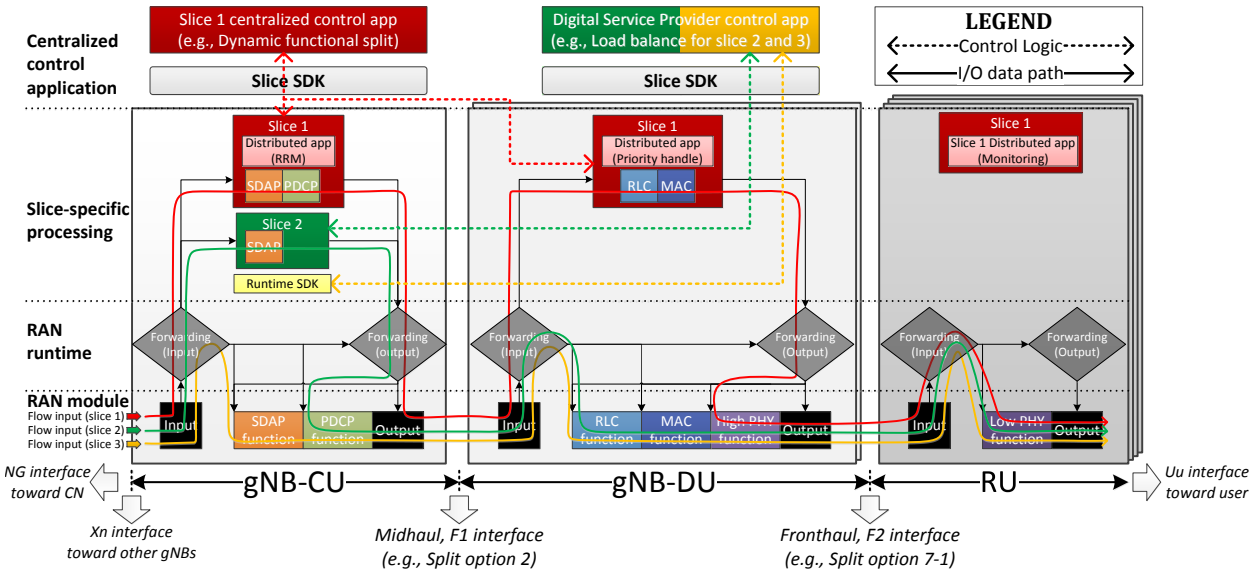


Fig. 3: An example of multi-service chaining and forwarding in a disaggregated RAN.

the underlying RAN module to enable each slice to control and manage its service by requesting radio resources, applying CLs, and accessing states. As for the runtime UP API, the protocol data units (PDUs) of the corresponding network layer are exchanged, possibly including some extra headers for flow-based match-action processing [9].

Finally, the dedicated control application can be either distributed or centralized, relying on (a) the runtime SDK to access the slice-specific resources, states, and processing, and/or (b) the data and API that are either slice-specific or open through the *slice SDK*. Such data may contain an abstracted network view, which will be further elaborated on in Section III. These centralized applications can be developed and deployed by the digital service provider that consumes the data and API from one or more network slices. For instance, the content optimization application in Fig. 2 can utilize the exposed RAN information from different slices to dynamically adjust the video bit rate.

B. Slicing in disaggregated RAN

Following the RAN evolution, a disaggregated RAN example is presented in Fig. 3 with the three-tier RAN nodes (gNB-CU, gNB-DU, RU) following the 1:m:n relationship. Hence, the slice service chain can be split between RU/gNB-DU (i.e., fronthaul with F2 interface [10]) and gNB-DU/gNB-CU (i.e., midhaul with F1 interface[1] identified in 3GPP TS38.470). Note that several other interfaces are identified

[1] It is surveyed as V1 interface for the LTE system in 3GPP TR37.876.

in 3GPP TS38.401: NG, Xn, and Uu interfaces are between the gNB and CN, another gNB, and the user equipment, respectively. The overall service function chain of each slice can be composed horizontally between RAN nodes (gNB-CU/gNB-DU/RU) and/or vertically when customized CP/UP processing is required tailored to service requirements. For instance, slice 1 customizes its SDAP, PDCP, RLC, and MAC, and slice 2 customizes its SDAP processing.

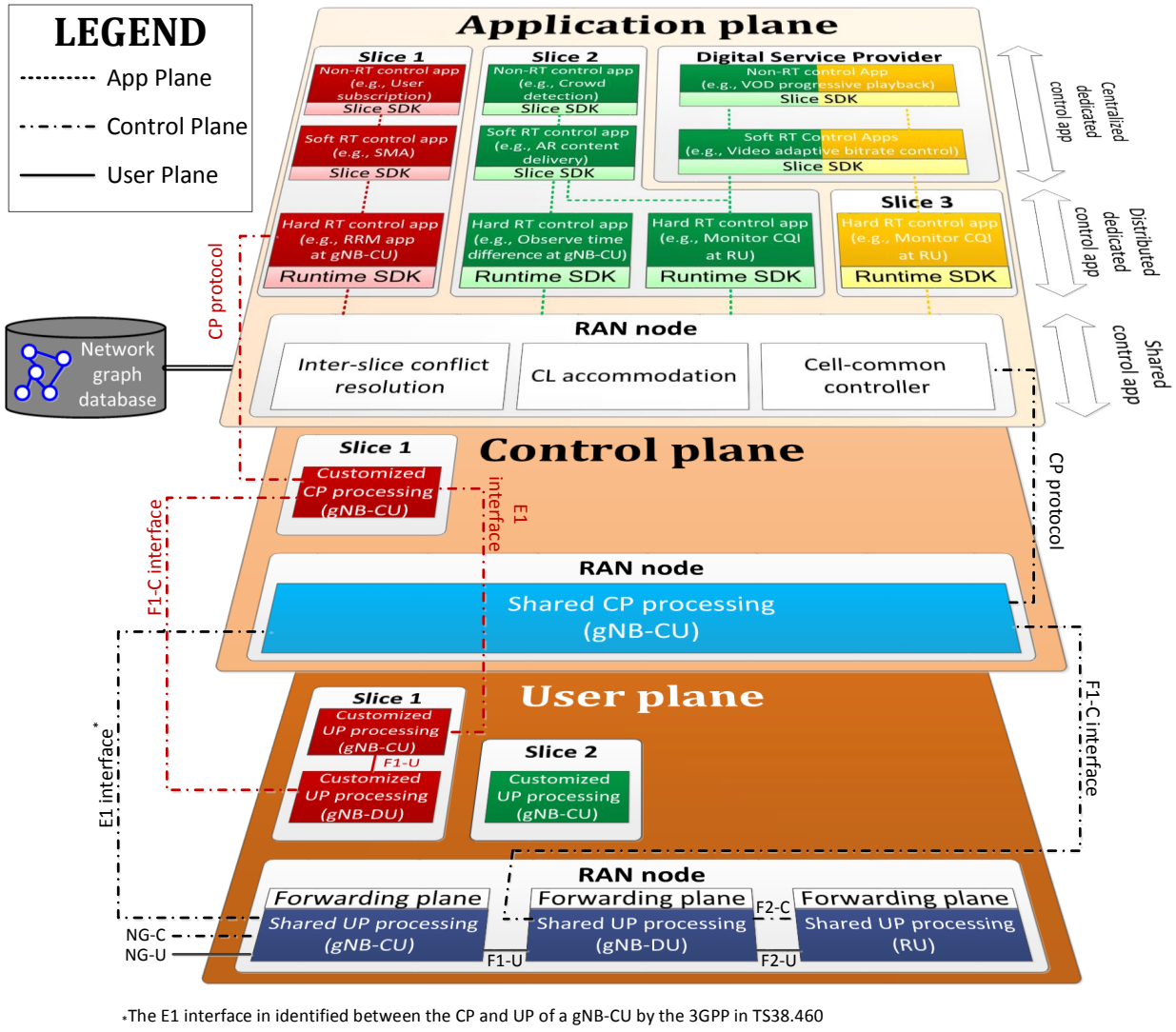
Based on the virtualized slice-specific view exposed from disaggregated RAN nodes, the control applications are facilitated to monitor underlying RAN information and to compose the CLs. Take slice 1 as an example. The RRM application at gNB-CU can manage the radio resource, the priority handling application at gNB-DU can dynamically prioritize the channels and users over its customized MAC processing, and the monitoring application at RU can provide real-time RAN information (e.g., the channel quality indicator [CQI]). Moreover, the centralized application of slice 1 can update the functional split between the customized processing at gNB-CU and gNB-DU via transporting the slice 1 data among multiple RAN runtime instances. Additionally, the application of a digital service provider can plug and play its CLs to the customized processing (slices 2) and shared processing (slice 3) leveraging the *slice SDK*.

III. Flexible and programmable RAN control

The RAN monitoring and control are performed by chaining dedicated and/or shared control applications, allowing a slice to flexibly (a) process CP/UP states to create and share knowledge and (b) apply actions to the underlying RAN modules. Specifically, in Fig. 4, two components are highlighted to support slice-specific control application developments and deployments: (1) Runtime SDK and *slice SDK* and (2) cross-domain control application chaining.

A. SDKs

Generally, an SDK provides a software development environment to simplify the design, development, testing and update of applications. It abstracts the underlying network by means of technology-agnostic and technology-dependent APIs and includes a group of libraries to provide specific functions and methods to be accessed through one or more API calls. Within the architecture depicted in Fig. 4, a two-level abstraction view of the underlying RAN entities is provided through both runtime SDK and *slice SDK*. First, the runtime SDK can expose both high- and low-level APIs. The high-level APIs rely on the RAN runtime services mentioned in Section II-A to allow for (1) creating/updating/destroying a virtual BS instance on top of RAN modules, (2) collecting monitoring metrics and KPIs for analytic purposes, and (3) retrieving/allocating the virtualized state and resource corresponding to a slice-specific network view.



*The E1 interface in identified between the CP and UP of a gNB-CU by the 3GPP in TS38.460

Fig. 4: The SDK and application plane for flexible and programmable RAN control.

In contrast, the low-level APIs utilize the aforementioned runtime CP/UP APIs to access the instantaneous network state information for a specific slice (e.g., the BS configuration and relevant user information), and to modify the CP/UP processing of the underlying RAN modules (e.g., the user measurement configuration). Moreover, the second-level abstraction is enabled by the specific *slice SDK* to facilitate the extensibility and coordination among control applications spanning different technology domains (e.g., RAN and CN) and administrative domains (e.g., communication and digital services providers), corresponding to open or slice-specific data and APIs, as shown in Fig. 2. Such a *slice SDK* can also expose context-aware semantic information [11] to facilitate the reasoning of actionable knowledge from heterogeneous information sources and foster interoperability among a variety of applications.

More specifically, we elaborate the following capabilities provided by the runtime SDK to enrich the advanced functionalities of control applications:

1) Authentication, authorization and management: Before utilizing any API calls, the application needs to be authenticated. Moreover, the slice-customized application must be authorized by the RAN runtime when accessing the slice-specific state information. Hence, it must provide its credential (or granted access token) and the identities of the target users and BSs for authorization. However, no authorization is needed when accessing the public BS information, such as a globally unique cell identity. Besides, a slice management operation can handle the life-cycle management of a network slice. It can also dynamically upgrade or downgrade the overall slice to a new profile, involving the related adaptations in terms of quality of service (QoS) control, PNF/VNF configuration, and so forth.

2) Metrics monitoring: This provides APIs for monitoring slice-related metrics and KPIs over multiple granularities according to the exposure level (e.g., resource, network function and application, slice, and service levels) and enables P&P applications for runtime control and adaptation. Furthermore, monitoring metrics of the corresponding cell/slice/user can be retrieved after authorization.

3) Control and delegation: The SDK allows the application of the control decisions over a cell/slice/user, depending on the slice-specific virtual network view and the VNF implementation. For example, two schedulers can have different parameter sets based on their functionalities and slice requirements. Moreover, the control delegation enables delegating the decision to others (e.g., the distributed application or RAN runtime), effectively reprogramming the underlying RAN modules.

4) Network graph database: This provides the graph-based primitives (e.g., split or merge) to operate on the network information, which can efficiently model, traverse and correlate more complex and dynamic relations between densely deployed RAN nodes. Moreover, it can naturally support the multi-tenant application through graph partitioning into subgraphs for multiple substrates (e.g., multi-domain or multi-service). Hence, each application can perform its graph-based operations (e.g., shortest path) that take node relationships in time series into account in its abstracted network view.

For instance, the network graph shown in Fig. 5 depicts three slices in a disaggregated RAN deployment. The disaggregated RU (i.e., RU_1 to RU_3), DU (i.e., DU_1, DU_2), and CU (i.e., CU_1) are virtualized for three different services to serve five users (i.e., u_1 to u_5). For instance, RU_1 is virtualized for slice 1 and slice 2 as $RU_{1,1}$ and $RU_{1,2}$, respectively. The edges between the vertices can represent their relations and be used for different applications. For example, the edges between a user and a virtualized RU, as in $u_1 \leftrightarrow RU_{1,2}$, can represent the measured CQI or traffic metrics. Such a subgraph can be utilized as input for handover or traffic-steering applications. Additionally, the edges between disaggregated RAN entities, as in $CU_{1,1} \leftrightarrow DU_{1,1} \leftrightarrow RU_{1,1}$, capture their association and the applied functional splits for multi-

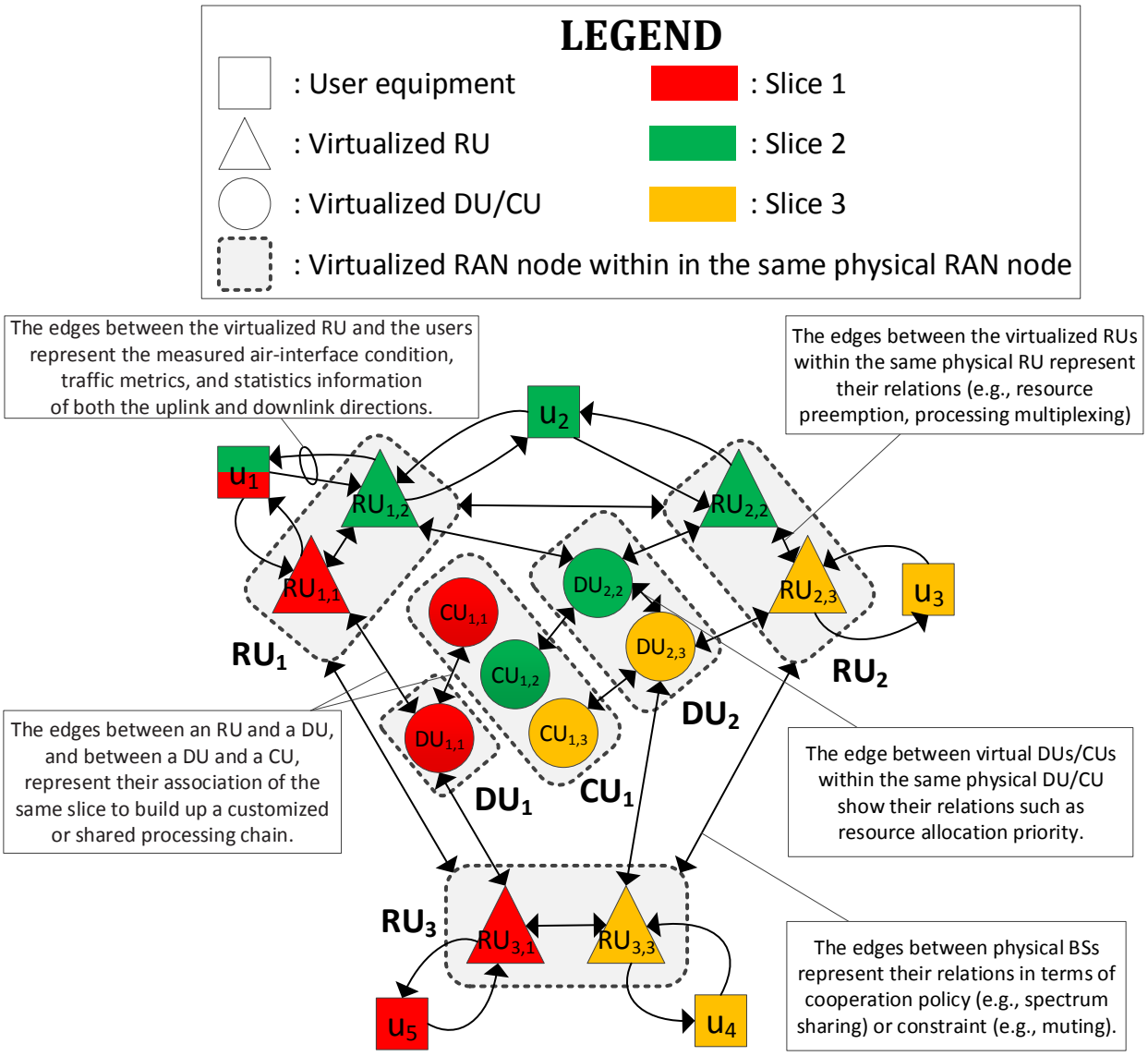


Fig. 5: An example of a network graph in a disaggregated RAN deployment.

service chaining and placement application. Furthermore, the edges between virtualized instances within the same physical RAN node, as in $RU_{2,2} \leftrightarrow RU_{2,3}$ and $DU_{2,2} \leftrightarrow DU_{2,3}$, show their relations in terms of sharing (e.g., multiplexing) and priority (e.g., preemption) that can be used for resource management, while the edges between different physical entities, as in $RU_2 \leftrightarrow RU_3$, depict the policy of cooperation (e.g., spectrum sharing) or constraint (e.g., exclusive muting) that can be combined with other graphs (topology graphs, for example) for dynamic radio spectrum management. In summary, the graph database can naturally represent complex relations and be partitioned or combined for control applications among multiple substrates.

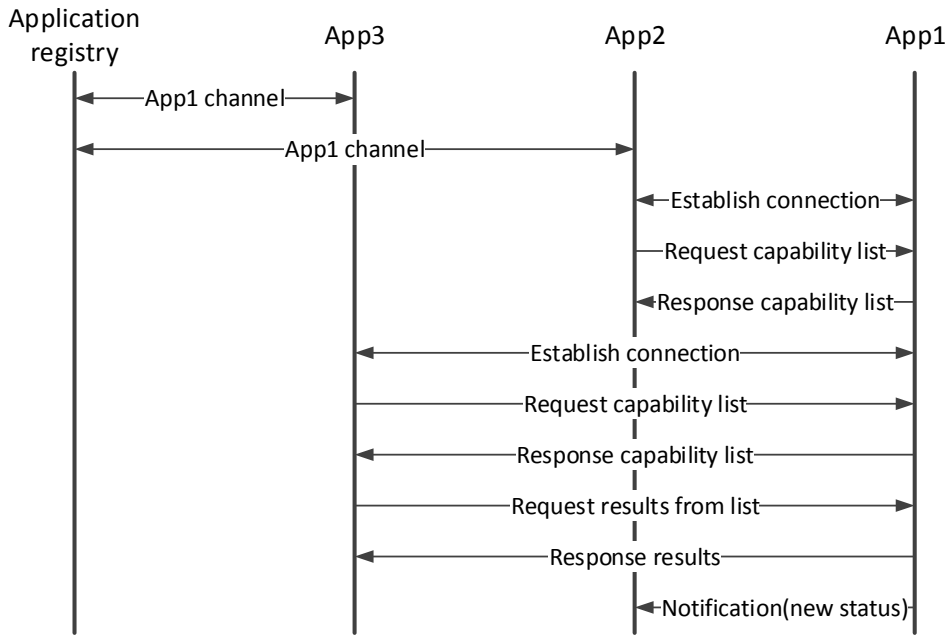


Fig. 6: The protocol flow for communications between control applications.

B. Single- and cross-domain application chaining

To produce sophisticated CLs in a multi-service environment, various dedicated/shared control applications can be chained together, as shown in Fig. 4, forming the application plane. Such chaining enables the automation and extendibility of the network control operations and improves the decision-making process across different slices based on common monitoring information. Specifically, three categories of applications can be chained: (a) non-real-time applications that enforce CLs when possible or when being instructed by higher layers, (b) soft-real-time applications that require an average delay guarantee within a tolerance when performing CLs, and (c) hard-real-time applications that require a delay guarantee when applying CLs, which otherwise cause a performance degradation.

Moreover, the communication between control applications can produce several benefits: it can (1) structure an application as a collection of loosely coupled micro-services [12], (2) synchronize applications' status when cooperation is needed, and (3) allow the implementation of a common interface for different applications. An example is shown in Fig. 6 with three applications, i.e., App1, App2, App3. Both App2 and App3 first register for the communication channel toward App1 and request the capability list from App1. Then, App3 asks for the latest results from the list, and thus App1 responds with the results and notifies App2 of the new status.

An example with three slice instances is provided in Fig. 4. For slice 1, the user's subscription information (e.g., user classes) can be utilized to allocate its carrier frequency and radio bandwidth

through the SMA and to control its radio resource through the RRM application in a single administrative domain (i.e., the same communication service provider). A cross-domain example is also shown in which the digital service provider can offer both a customized video-on-demand (VOD) playback application and a soft-real-time video adaptive bit rate (ABR) control based on the monitored CQI from one or more slices (e.g., slices 2 and 3). Note that such an application plane can span several disaggregated RAN nodes, e.g., slice 2 can monitor the CQI at RU and the observed time difference at gNB-CU for augmented reality content delivery and crowd detection applications.

IV. Proof of Concepts

To explore the potential of chaining control applications over the application plane, we implemented an LTE-based prototype of RAN runtime slicing system based on the OpenAirInterface [13] and Mosaic5G [14] platforms. We created remote slices using an asynchronous communication channel toward the RAN runtime, embedded its control applications, and operated on virtualized resources and states based on the provided runtime SDK. In following, we present the results for two use cases: (a) RAN-aware video optimization and (b) Subscription-aware RAN resource provisioning.

A. RAN-aware video optimization

We adapt the video bitrate based on instantaneous RAN information (i.e., the CQI), relying on the chaining of the monitoring and ABR applications, following the steps shown in Fig. 7. Hence, video segments of different qualities were provided based on the requested rate mapped from the CQI value to maintain the video QoE [15]. We first measure the maximum user good-put corresponding to different CQI values in both the downlink and uplink directions shown on the top of Fig. 8. Afterwards, two experiments were conducted to compare the cases with and without RAN-aware video optimization, i.e., labeled ABR and Fix respectively, when varying the CQI from 4 to 15. The former showed that the video quality[2] can be adapted from WQVGA, WVGA, WSVGA, HD to FHD, when the user channel quality is improved with fewer dropped video frames. However, the latter used fixed HD video quality irrelevant to the CQI fluctuation. We can observe that a large amount of dropped frames and zero buffer length when the average CQI is 4 and an inferior video quality when the CQI is 15. The results revealed that the QoE can be significantly improved when chaining monitoring and ABR applications that belong to two different administrative domains.

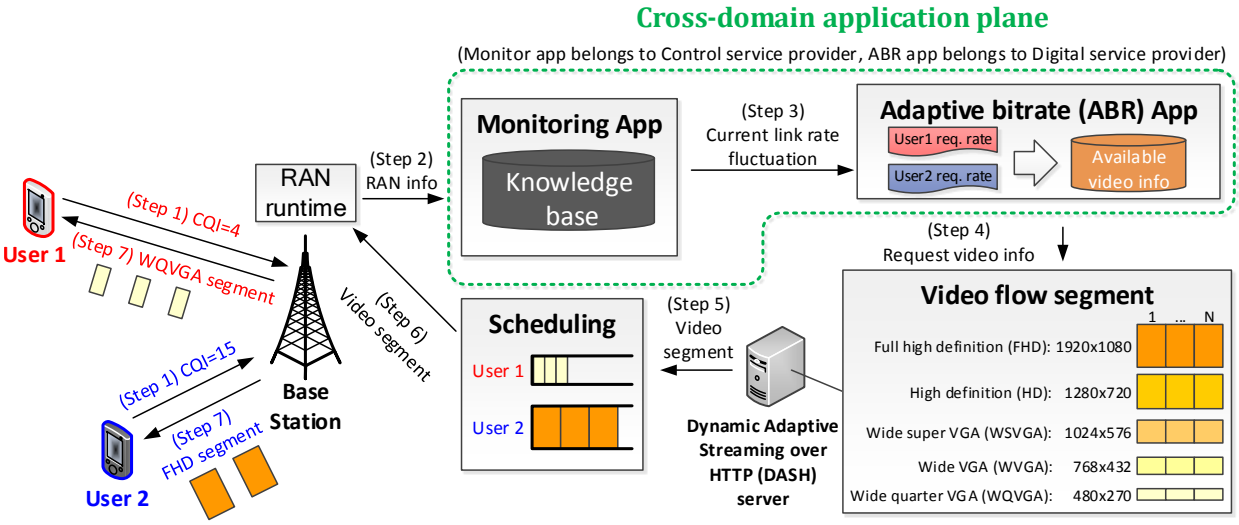
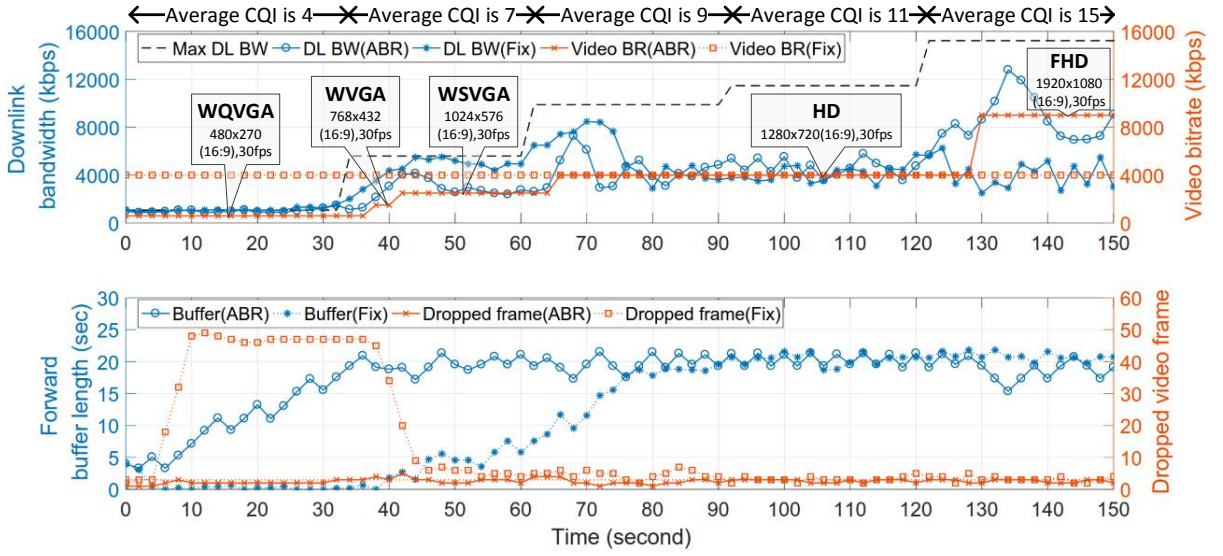


Fig. 7: The process flow of RAN-aware video optimization use case.

CQI	Downlink (Mbps)	Uplink (Mbps)
4	1.08	0.69
7	5.59	2.49
9	9.88	4.47
11	11.47	6.04
15	15.22	8.08



Acronym

WQVGA : Wide Quarter Video Graphics Array
WVGA : Wide Video Graphics Array
WSVGA : Wide Super Video Graphics Array
HD : High Definition
FHD : Full High Definition

Fig. 8: The measured maximum good-put of corresponding CQI values (top) and experiment results consisting downlink bandwidth, video bitrate, buffer length and dropped frames (bottom).

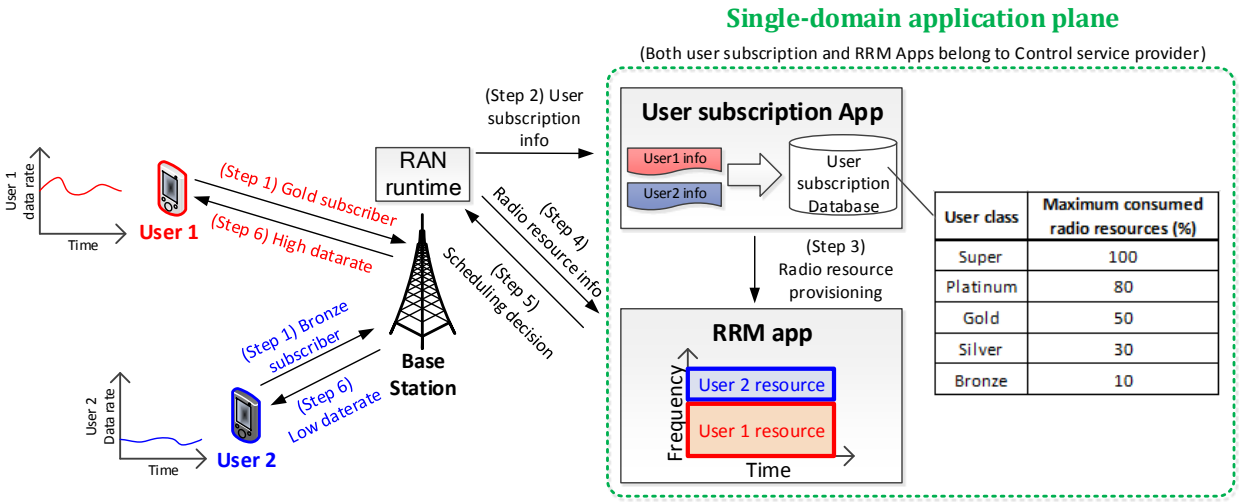


Fig. 9: The process flow of a subscription-aware RAN resource provisioning use case.

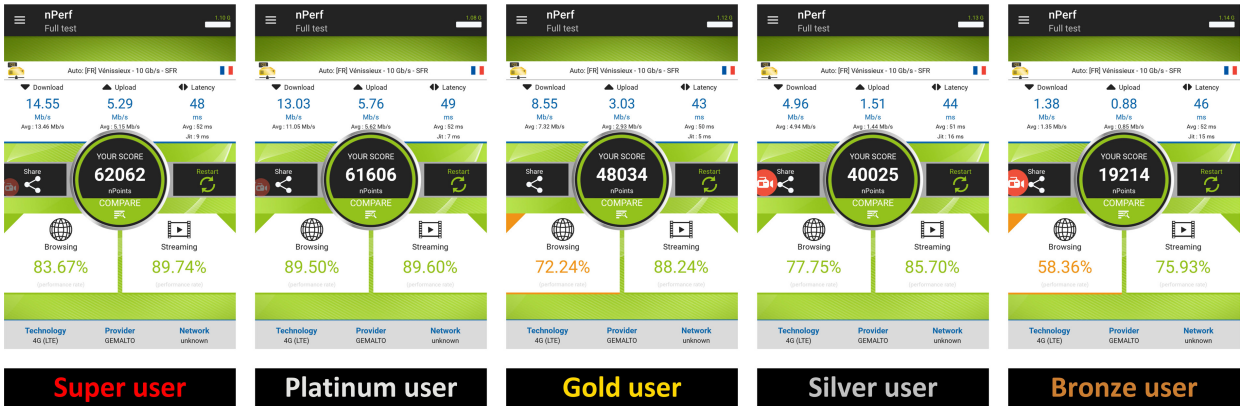


Fig. 10: From left to right, the measured QoE for five user classes: Super, Platinum, Gold, Silver and Bronze.

B. Subscription-aware RAN resource provisioning

We then chained slice subscription and RRM applications within a single administrative domain to provision different radio resources for different user classes, following the steps depicted in Fig. 9. These five user classes have different pre-defined policy profiles that determine the amount of radio resources a user can consume in the downlink and uplink directions. To quantify the overall QoE of different user classes, the neprf application[3] was installed on the commercial-off-the-shelf user equipment connecting to the OpenAirInterface BS, as shown in Fig. 10. The super user got the best score, considering bit rates, delay and jitter, web browsing and video streaming performance rate. The Platinum users score was close

[2] https://wikipedia.org/wiki/Graphics_display_resolution [accessed on July-13-2018] [3] https://www.nperf.com/en/ [accessed on 13-Jul-2018]

to the super user's, while there were performance drops for the gold and silver users in web browsing and bit rate. The bronze users suffered from significant drops in the bit rate, web browsing, and video streaming. Summing up, chaining both user subscription and RRM applications can provision RAN radio resource in real-time.

V. Conclusions

In this article, we proposed the RAN runtime slicing system to enable a slice-friendly development environment and provided the runtime SDK for the development of control applications. To enable flexible CL programmability, we proposed the two-level abstraction concept, relying on several SDK capabilities and the application plane to chain shared/dedicated control applications. Finally, two use cases were presented over the proposed runtime SDK and RAN runtime to provide sophisticated and customized CLs when chaining control applications.

Acknowledgement

This work has received funding from the European Union's Horizon 2020 framework program under grant No. 671639 (COHERENT), No. 762057 (5G-PICTURE) and No. 761913 (SliceNet).

References

[1] P. Marsch, I. Da Silva, O. Bulakci, M. Tesanovic, S. E. El Ayoubi, T. Rosowski, A. Kaloxylos, and M. Boldi, "5G Radio Access Network Architecture: Design Guidelines and Key Considerations," *IEEE Communications Magazine*, vol. 54, no. 11, pp. 24–32, Nov. 2016.

[2] X. Foukas, N. Nikaein, M. M. Kassem, M. K. Marina, and K. Kontovasilis, "FlexRAN: A Flexible and Programmable Platform for Software-Defined Radio Access Networks," in *Proceedings of the 12th International on Conference on Emerging Networking EXperiments and Technologies (CoNEXT '16)*. ACM, 2016, pp. 427–441.

[3] X. Foukas, M. K. Marina, and K. Kontovasilis, "Orion: RAN Slicing for a Flexible and Cost-Effective Multi-Service Mobile Network Architecture," in *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking (MobiCom '17)*. ACM, 2017, pp. 127–140.

[4] C.-Y. Chang and N. Nikaein, "RAN Runtime Slicing System for Flexible and Dynamic Service Execution Environment," *IEEE Access*, vol. 6, pp. 34 018–34 042, Jun. 2018.

[5] N. Nikaein, E. Schiller, R. Favraud, K. Katsalis, D. Stavropoulos, I. Alyafawi, Z. Zhao, T. Braun, and T. Korakis, "Network Store: Exploring Slicing in Future 5G Networks," in *Proceedings of the 10th International Workshop on Mobility in the Evolving Internet Architecture (MobiArch '15)*. ACM, 2015, pp. 8–13.

[6] R. Riggio, M. K. Marina, J. Schulz-Zander, S. Kuklinski, and T. Rasheed, "Programming Abstractions for Software-Defined Wireless Networks," *IEEE Transactions on Network and Service Management*, vol. 12, no. 2, pp. 146–162, Jun. 2015.

[7] K. Katsalis, N. Nikaein, E. Schiller, A. Ksentini, and T. Braun, "Network Slices toward 5G Communications: Slicing the LTE Network," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 146–154, Aug. 2017.

[8] C.-Y. Chang, N. Nikaein, R. Knopp, T. Spyropoulos, and S. S. Kumar, "FlexCRAN: A Flexible Functional Split Framework over Ethernet Fronthaul in Cloud-RAN," in *2017 IEEE International Conference on Communications (ICC)*, May 2017, pp. 1–7.

[9] P. Bosshart, G. Gibb, H.-S. Kim, G. Varghese, N. McKeown, M. Izzard, F. Mujica, and M. Horowitz, "Forwarding Metamorphosis: Fast Programmable Match-action Processing in Hardware for SDN," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4, pp. 99–110, Aug. 2013.

[10] A. Sutton, "5G Network Architecture," *Journal of The Institute of Telecommunications Professionals*, vol. 12, no. 1, pp. 9–15, 2018.

[11] A. Al-Saadi, R. Setchi, and Y. Hicks, "Semantic Reasoning in Cognitive Networks for Heterogeneous Wireless Mesh Systems," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 3, pp. 374–389, Sep. 2017.

[12] N. Dmitry and S.-S. Manfred, "On Micro-services Architecture," *International Journal of Open Information Technologies*, vol. 2, no. 9, pp. 24–27, 2014.

[13] N. Nikaein, M. K. Marina, S. Manickam, A. Dawson, R. Knopp, and C. Bonnet, "OpenAirInterface: A flexible platform for 5G research," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 5, pp. 33–38, Oct. 2014.

[14] N. Nikaein, C.-Y. Chang, and K. Alexandris, "Mosaic5G: Agile and flexible service platforms for 5G research," *ACM SIGCOMM Computer Communication Review*, vol. 47, no. 3, Jul. 2018.

[15] X. Xie, X. Zhang, S. Kumar, and L. E. Li, "piStream: Physical Layer Informed Adaptive Video Streaming Over LTE," in *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking (MobiCom '15)*. ACM, 2015, pp. 413–425.

Chia-Yu Chang received the B.S and M.S. degrees from National Taiwan University, Taiwan, in 2008 and 2010, respectively. He is currently pursuing the Ph.D. degree with Communication Systems Department, EURECOM, France. Between 2010 and 2015, he was in MediaTek Inc., Taiwan, as a senior engineer for the design of 3G/4G cellular communication system architecture and algorithm design. He participates in several collaborative research projects related to the 5G communication system architecture and protocol design in EU Horizon 2020 framework programs. His research interests include design for communication system architecture, wireless network virtualization, and cross-layer algorithm.

Navid Nikaein received the Ph.D. degree (docteur ès sciences) in communication systems from the Swiss Federal Institute of Technology EPFL in 2003. He is currently a Tenured Associate Professor with the Communication Systems Department, EURECOM, France. He is also leading a group focusing on 4G-5G experimental system research related to radio access and core networks with a blend of communication, cloud computing, and data analysis. Broadly, his research contributions are in the areas of wireless access layer techniques, networking protocols and architectures, service-oriented RAN/CN following SDN, NFV, MEC design principles, and wireless network prototyping and emulation/simulation platforms. He has a proven track record in collaborative research projects related to 4G-5G and beyond in the context of European FP6, FP7, and H2020 framework programs, and served as a Project Manager, a Technical Coordinator, and a Work Package Leader. He is also leading the development of the radio access layer of OpenAirInterface and its evolution towards 5G as well as coordinating the Mosaic5G initiative whose goal is to provide software-based 4G/5G service delivery platforms.