Slicing and Orchestration in Service-Oriented RAN Architecture

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

I. Introduction

Radio access network (RAN) slicing is one of the key enablers to realize the service-oriented 5G vision and deliver RAN-as-a-service. To dynamically manage and orchestrate diverse slices [1], a *multi-service execution environment* is required to enable on-the-fly virtualization of RANs on the top of the same physical RAN infrastructure, each customized and programmed as per slice requirements. The underlying RAN infrastructure can be either monolithic or *disaggregated*, where RAN is decomposed into radio unit (RU), distributed unit (DU) and centralized unit (CU) with flexible functional split among them [2]. Such disaggregation provides the capability to *abstract* resources, states and function modules in order to be reusable and composable across slices, while retaining the centralization benefits for coordinated and cooperative processing. With abstraction, different levels of isolation and sharing among resource and state can be provided. Additionally, slice functions can be customized in terms of user-plane (UP) and control-plane (CP) processing to reflect slice-specific requirements with the access to a portion of resources and states in a virtualized form.

Several 3GPP studies are conducted to address the challenges of end-to-end (E2E) network slice management and orchestration [3], and RAN slicing [4]. Additionally, various ETSI NFV-MANO architectural framework implementations like open source MANO [1], OPNFV [2], and ONAP [3] are addressing the management and orchestration of network slices for next generation network. Nevertheless, RAN slicing remains challenging in providing different levels of resource and state isolation and sharing while enabling slice orchestration for multi-service RAN infrastructures. This calls for a unified and flexible execution environment to run multiple virtualized RAN instances with the required level of customization over the monolithic or disaggregated RAN. To this end, we propose a RAN slicing runtime system to facilitate slicing and orchestration of RAN [5].

[1] osm.etsi.org [2] www.opnfv.org [3] www.onap.org

II. RAN SLICING RUNTIME SYSTEM

The RAN slicing architecture is shown in Fig. 1 with the RAN runtime being the core component by which each slice interacts with underlying RAN module to access resources and states, and control the behaviors.

A. RAN runtime

Within the RAN runtime, three services are provided: (a) **slice manager**, (b) **virtualization manager**, and (c) **context manager** are operating over a shared **slice data**. Slice data includes both slice context (e.g., basic information to instantiate a slice service like its identity, user context and their slice association) and module context (e.g., CP and UP state information, module primitives) that are used to customize and manage a slice in terms of the required resources, states, processing, and users. Slice data can be transferred or shared among different runtime instances based on user and network dynamics such as user handover and/or RAN service template change when updating functional splits [2].

Based on the slice service template and slice context, the slice manager determines the CP/UP processing chain for each slice and each traffic flow, and programs the **forwarding plane** allowing to direct the input and output streams across multiplexed processing operated by the underlying RAN module and customized processing performed by each slice. An E2E RAN service is operated by the slice manager in support of service continuity when slice service template is updated. Further, slice manager is responsible for taking actions based on a set of policy rules when detecting any conflicts among multiple slices.

The virtualization manager provides the required level of isolation and sharing among slices. Specifically, it partitions resources and states, abstracts physical resources and states to/from the virtual ones, and reveals virtual resources and states to a slice, which are decoupled from the physical ones. Regarding resource aspect, it can rely on the two-level scheduler framework that abstracts Physical Resource Blocks (PRBs) into independent virtual Resource Blocks (vRBs) scheduled by slice customized scheduler [6]. Once vRBs are allocated, the virtualization manager maps the vRB/users allocation to PRB/users allocation by considering the slice priority and service level agreement (SLA).

The context manager performs CRUD operations (i.e., create, read, update, and delete) on both slice and module context. To create a slice context, it firstly performs slice admission control based on the service template that defines the required processing, resources, and states. Upon admission control, module context is used to register (a) slice-specific life-cycle primitives to the slice manager, and (b) requested resources and/or performance to the virtualization manager. At this stage, a slice starts to consume the runtime services.

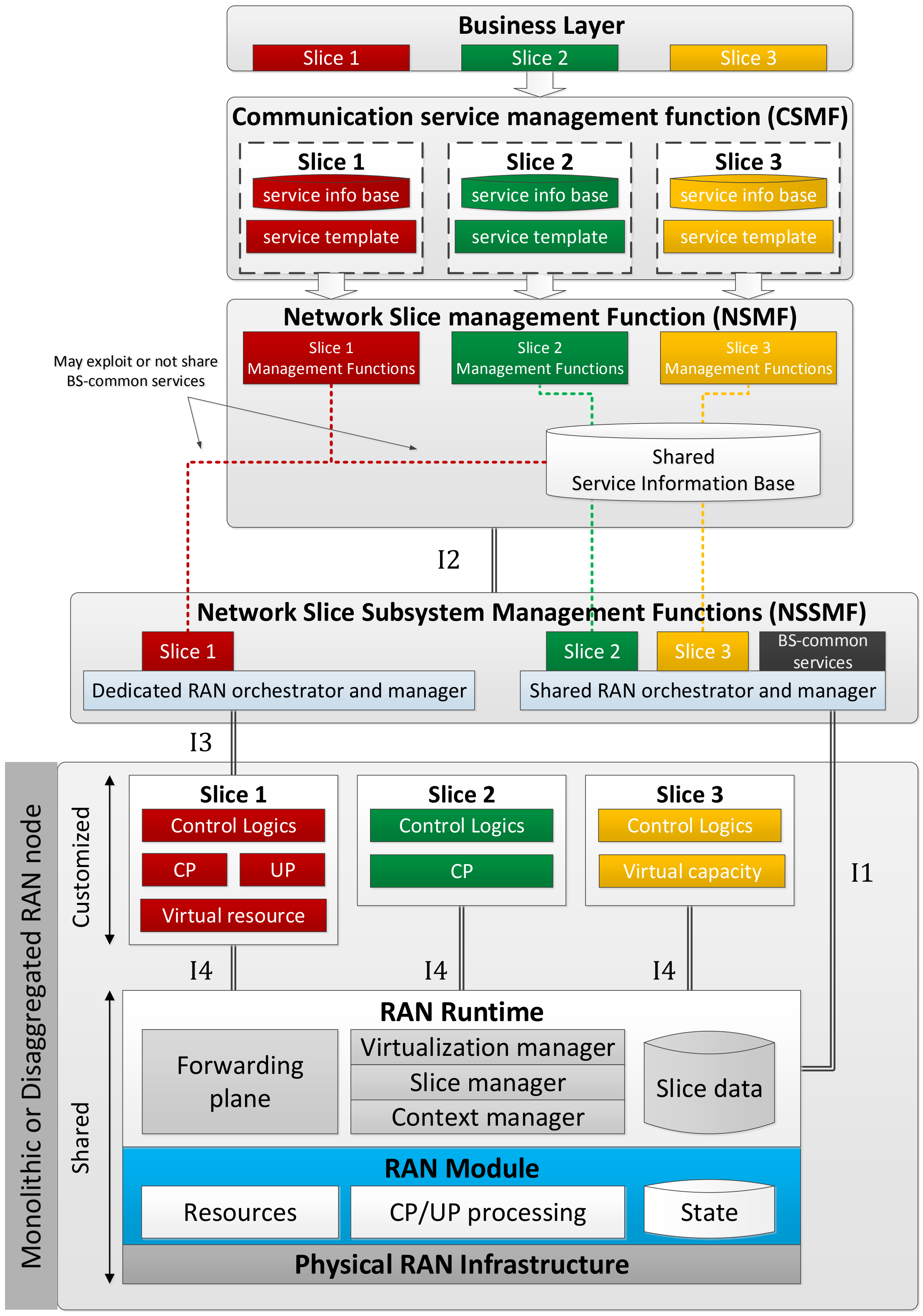

Fig. 1: RAN slicing runtime system architecture.

B. Slice Orchestration

The RAN runtime provides a multi-service environment with four interfaces (`I1` to `I4` in Fig. 1) facilitating slice orchestration and management and complying with communication service management function (CSMF), network slice management function (NSMF) and network slice subnet management function (NSSMF) defined by 3GPP [3].

`I1` interface serves several purposes. Firstly, information about the activated services and capabilities of runtime are exposed through such interface. Secondly, it serves the service registration to runtime during slice creation based on the service template. Thirdly, monitoring and feedback messages are retrieved to facilitate slice coordination and service template update.

`I2` interface is between the NSMF and NSSMF. Here, a need for dedicated service orchestration and management is highlighted to facilitate the slice owner to fully control its service over deployed RAN nodes with several merits. It enables applying slice-specific life-cycle primitives, self-maintaining service continuity and isolation, and allowing for control logics being applied for slice-specific applications, e.g., slice-dependent load balancing.

`I3` interface is between the dedicated service management and corresponding slice at RAN node through which a slice owner can customize service monitoring and management as per slice needs. Further, it is used by the slice manager of runtime to indicate any changes in the underlying RAN infrastructure and/or RAN module allowing the dedicated service manager to adapt its service accordingly.

`I4` interface provides communication channels between slices and runtime allowing each slice as a separate process, whether being local or remote. Hence, each slice is executed isolatedly, either at host or guest level, leveraging OS and virtualization technologies like container or virtual machine.

C. Multi-service chaining

A slice service chain can be split between disaggregated RAN entities according to the functional split options defined by 3GPP [4]. Such chain can be composed horizontally based on the multiplexed functions provided by the underlying RAN modules, and/or vertically when customized CP/UP processing is required tailoring to service requirements. The overall service chain of each slice is composed by the forwarding plane in runtime, which can leverage SDN-based match-action abstractions to build slice forwarding input and output paths across multiplexed and customized functions. An example of a disaggregated RAN with different levels of customization in a downlink path is shown in Fig. 2.

Note that slice-specific function customization is described in the service template, and the customized forwarding path of each slice are managed by the slice manager of runtime in coordination with slice orchestrator. As the runtime maintains the states for both customized and multiplexed network functions

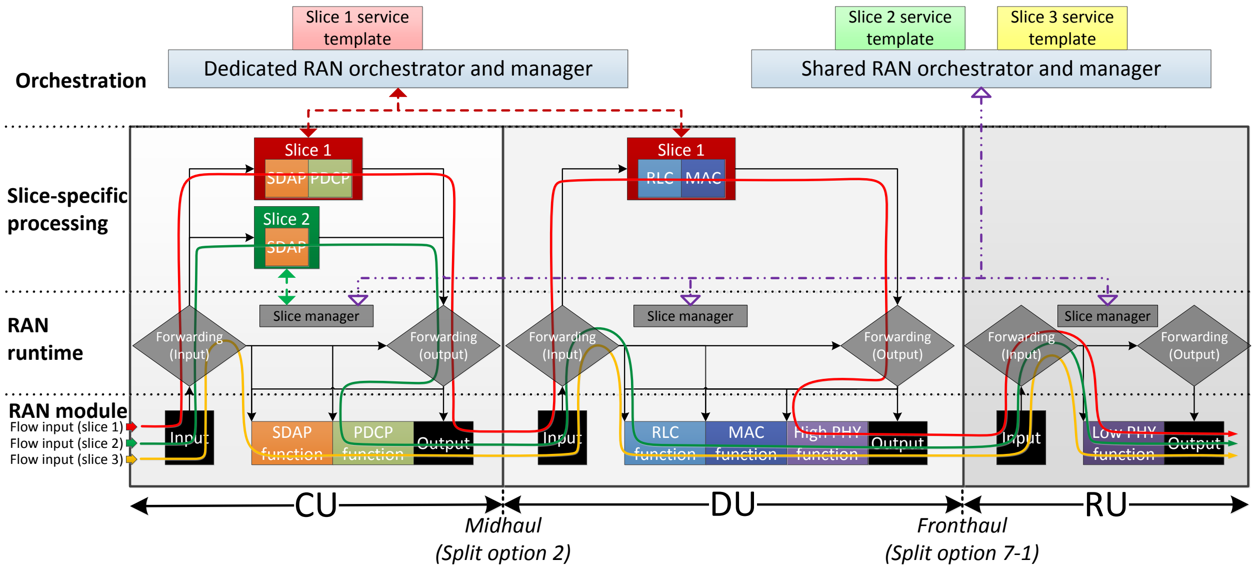

Fig. 2: An example of multi-service chaining in disaggregated RAN.

within the slice data, it facilitates dynamic service template update by handling state synchronization among involved RAN entities.

D. Examples of RAN slicing

We present three different slices shown in Fig. 1, and each slice maintains its service information base (SIB) and service template. Several information can be retrieved through SIB: SLA, user information, user to slice association, and service inventory. The service template contains tangible per-slice requirement through CSMF functionality. Using such template, the NSMF prepares network service instances, in which each slice is orchestrated and managed in shared or dedicated NSSMF at RAN-domain.

Take slice 1 as an example, it relies on a dedicated service orchestrator and manager such that the resulted slice are fully controlled by the slice owner with customized CP/UP processing and virtual resources. While slice 2 and 3 do not request such customization, thus their services are managed through a shared RAN-domain service orchestrator based on the information stored in a shared SIB.

E. Summary

A 3GPP-compatible RAN slicing and service orchestration architecture is provided exploiting RAN runtime to enable a multi-service execution environment in disaggregated RAN. Under 5G-PPP research program, both 5G-PICTURE and SLICENET projects are exploring further challenges. For more information, please visit www.5g-picture-project.eu and slicenet.eu, respectively.

REFERENCES

[1] *IMT-2020 Deliverables*, ITU-T Focus Group, 2017.

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

[3] *TR 28.801 Study on management and orchestration of network slicing for next generation network (Release 15)*, 3GPP, Sep. 2017.

[4] *TR 38.801 Study on new radio access technology: Radio access architecture and interfaces (Release 14)*, 3GPP, Mar. 2017.

[5] C.-Y. Chang and N. Nikaein, "RAN slicing runtime system for flexible and dynamic service execution environment," Tech. Rep., 10 2017.

[6] A. Ksentini and N. Nikaein, "Toward Enforcing Network Slicing on RAN: Flexibility and Resources Abstraction," *IEEE Communications Magazine*, vol. 55, no. 6, pp. 102–108, 2017.

Navid Nikaein is an assistant professor in Communication System Department at EURECOM. He received his Ph.D. degree in communication systems from the Swiss Federal Institute of Technology EPFL in 2003. He is leading a group focusing on experimental system research related to radio access and core networks with a blend of communication, cloud computing, and data analysis. He has led the development of the radio access layer of OpenAirInterface and coordinating a new ecosystem of open-source projects, under Mosaic5G.io framework, to deliver 5G as software-based service platforms.

Chia-Yu Chang is currently a Ph.D. candidate in Communication System Department at EURECOM since 2015. He participates in collaborative research projects related to future communication system architecture and protocol design in several H2020 framework programs. He received the B.S.E.E. and M.Sc. degrees from National Taiwan University, Taiwan, in 2008 and 2010, respectively. Prior to joining EURECOM, he had worked as a senior engineer in MediaTek Inc. and been responsible for the 3G/4G baseband communication system architecture and algorithm design.