Reference architecture for slicing in LoRAWAN networks (updated version)

Fabrice Guillemin, Alessandro Aimi, Tanguy Kerdoncuff, Stephane Rovedakis

Reference architecture for slicing in LoRAWAN networks (updated version)

Project Deliverable D.1.1.2

INTELLIGENTSIA project

Executive summary

This document provides an update of the functional architecture designed for the Intelligentsia project. With regard to the previous deliverable D1.1.1, we pay special attention to the virtualisation of LoRaWAN Network Servers (LNS) and to the functional blocks which should be added in order to perform quality differentiation in LoRaWAN. Finally, we advocate for business driven approach to slicing leading to put some functions out the orchestrator platform in order to have more freedom in the instantiation of slices and to have a dedicated monitoring as in it is done today for Virtual Private Networks (VPNs) by supervision centers of operational networks.

Authors: Fabrice Guillemin, Alessandro Aimi, Tanguy Kerdoncuff, Stéphane Rovedakis

Revised by: All members of the Intelligentsia project.

Approved by: Fabrice Guillemin

Date: September 23, 2022

Contents

List of Figures

Acronyms

CDP Connected Device Platform.
CSMF Communication Service Management Function.

KPI Key Performance Indicator.

LNS LoRaWaN Network Server.

NEF Network Exposure Function.
NSMF Network Slice Management Function.
NSSI Network Slice Subnet Instance.
NSSMF Network Slice Subnet Management Function.
NWDAF Network and Data Analytics Function.

PDR Packet Delivery Ratio.
PoP Point of Presence.

QoE Quality of Experience.

RAN Radio Access Network.

SDN Software Defined Network.
SLA Service Level Agreement.

VNF Virtualized Network Function.
VPN Virtual Private Network.

Chapter 1

Introduction

The objective of the Intelligentsia project is to develop the concept of slicing in the context of LoRaWAN. This is a very ambitious objective provided that the LoRa technology is basically best effort, relying on an ALOHA-like medium access protocol. However, with the wide spread of connected objects, many companies start designing business cases on the basis of LoRaWAN. The attractiveness of LoRa networks merely comes from their low cost deployment as well as the cheapness of devices. Operators of such networks, including the traditional telco operators such as Orange, are urged to propose the deployment of LoRa network with quality and reliability guarantees.

A method of offering differentiated quality and reliability in networks is to introduce the concept of slicing, provided that not all services require quality guarantees. In that case, an actor (for instance a service provider) negotiates with the network operator the setup of slice; the corresponding devices are then members of the slice on the basis of a subscription (for instance by using their MAC address).

The concept of slice covers various aspects:

- Performance: objectives in terms of packet loss (or equivalently Packet Delivery Ratio (PDR)), latency, reliability, etc.
- Customization: specific features in the treatment of packets transmitted or received by devices of a slice.
- Security: isolation of the treatment of packets, dedicated functions, etc.
- Assurance: via monitoring, the network operator can verify that the Service Level Agreement (SLA) negotiated with the client is respected, for instance by exposing some Key Performance Indicators (KPIs).

To meet these requirements, operators have various technical enablers, notably virtualization and resource management. The latter aspect concerns more specifically the air interface, which is the most critical with regard to quality assurance; the quality offered by the backhaul network connecting the radio gateways to the LoRaWaN Network Servers (LNSs) is a traditional dimensioning issue. There are also resource management issues regarding the virtualization infrastructure. We consider however in the following that these are second order issues and we shall focus on resource management issues for the air interface.

The data collected for monitoring and resource allocation are specified in WP2 of the Intelligentsia project. The resource management algorithms for the air interface are designed in WP3. A preliminary functional architecture has been proposed in Deliverable D1.1.1. In the present document, we give more details (especially with regard to the virtualization of LNSs and refine some aspects of the architecture.

In the proposal document, it was envisaged to virtualize the radio gateway, in the same way as gNodeBs are virtualized in the context of cellular networks, giving rise to the concept

of cloud Radio Access Network (RAN) (see [4] for instance). In the context of LoRaWAN, this would translate as follows. In Figure 1.1, we present what would be the traditional way of creating a Virtualized Network Function (VNF) from a LoRaWAN gateway. By replacing hardware with software, some radio processing functions can be moved to the cloud. In such a setup, the received radio signal is streamed over the infrastructure network up to the software brick responsible for its processing. This will however prove impossible to implement in the framework of the Intelligentsia project due to the proprietary nature of LoRaWAN gateways. This option will not be considered further in the Intelligentsia project. Nevertheless, since some algorithms designed in WP3 envisage radio management functions, some additional modules will be introduce to implement resource management schemes, even if radio gateways are left unchanged.

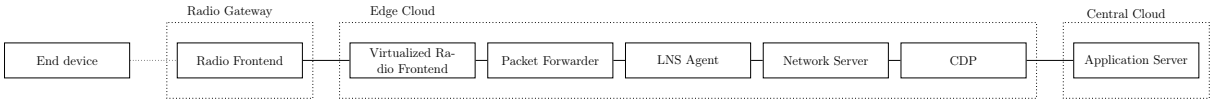

Figure 1.1: Virtualized LoRa radio function

This report is organized as follows: In Chapter 2, we introduce the various elements of the architecture. In Chapter 3, we provide an update of the functional architecture described in Deliverable D1.1.1. Some concluding remarks are presented in Section 4.

Chapter 2

Virtual LNS

The LNS is a key component of LoRaWAN, which enables connectivity, management, and monitoring of devices, gateways and end-user applications. The objectives of this function are to ensure the security, scalability and reliability of data routing throughout the network.

An LNS is composed of several functions (see Deliverable D1.1.1 for details):

The Network Server: This component handles all the LoRaWAN protocolar aspects, such as incoming and outgoing LoraWan frames or signaling.

Connected Device Platform (CDP): This function is also called Application Server in chirpstack and in some Semtech representations, and is the interface to the state-full data of the platform (stored in a database), such as the list of registered devices, their session keys, radio and application profile, etc.

The Join Server: This function interacts with the Network Server when a device requires access to the network, and will be the element authenticating the device, but will only share with the Network Server the keys that will let it route the LoRaWAN packet, but not the payload decryption one.

In this chapter, we describe how LNSs could be virtualized and the various options, which could be envisaged in the framework of slicing.

2.1 Network function in edge cloud

In a first step, we show how some functions of LoRaWAN could be moved to the edge of the network. Generally speaking, the cloud infrastructure of network operators tend to a three level architecture:

- The centralized cloud, which is at the national level and hosting applications very resource consuming in terms of resource or tolerant to delays, so that the round trip time through a nation wide network does not alter the Quality of Experience (QoE) of the users of the application.

- The fog cloud at the regional level, which roughly corresponds to the Point of Presence (PoP) level in the current infrastructures of networks. This level is in general the first IP point with routable IP addresses. The fog level is adapted to content delivery as storing content as this level prevents from multiple transfers of the same data through the national IP network. In addition, fog clouds can host time sensitive applications as the round time between end users and fog clouds is of the order of a several tens of milliseconds.

- Edge cloud is the cloud platform which is the closest to end users and should host applications which are very sensitive to delays. Edge clouds can host applications of the telco operator (e.g., cloudRAN) and those of end users (e.g., cloud gaming, etc.).

In the following, we consider edge cloud but fog cloud could also be envisaged; we do not consider time sensitive applications, where a device interacts with the infrastructure and hence, the location of virtualized components is more flexible when compared with use cases such as cloudRAN. The LNS is a good candidate for moving some functions closer to radio gateways.

In Figure 2.1, we present a possible approach to moving gateway side functions to the edge cloud. This specifically targets the packet forwarder code running on each gateway. This software gateway component is responsible for reading frames from the radio front-end and transmitting this upwards using an UDP protocol defined by Semtech. This provides a first layer of abstraction by unpacking the frontend output into LoRaWAN packet objects. One could therefore argue that this would be a valid candidate for migration cloud side, by moving the UDP packet generation to the cloud.

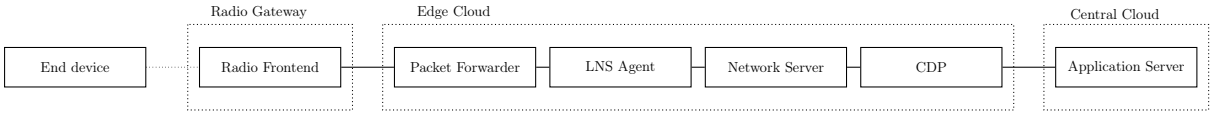

Figure 2.1: Edge side LoRa packet forwarding

We however do not, at this point, plan to explore this possibility. First, this software brick is already lightweight as is specifically targets gateways and is usually packaged and built by the gateway manufacturers themselves. Furthermore, this being a software component limits the cost benefits of migrating it. Lastly, other approaches detailed below will rely on a minimal packet awareness on the gateways in order to more efficiently balance the load of LNS packets between their destinations.

2.2 Gateway sharing between multiple LNSs

Figure 2.2 presents how a pool of gateways can be shared amongst multiple LNSs. In order to do so, the UDP stream generated by the gateway's packet forwarder needs to be split between all the gateway agents present on the gateway. Each agent is part of a given LNS, therefore enabling the coverage of many different device fleets (namely, members of slices in the context of the present document) using a common pool of gateways.

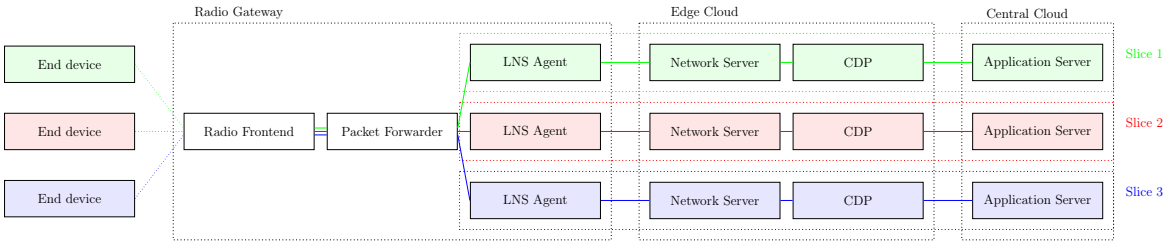

Figure 2.2: Gateway sharing between LNSs

This proposal can easily be implemented by using a UDP stream splitter, which requires some specific code. This however has some drawbacks in terms of scalability. While the agent duplication on a gateway could be addressed by allowing an agent to serve multiple LNS, the traffic of all the gateways is still duplicated to all the LNSs, which may get overloaded when the gateway pool grows.

2.3 Gateway sharing through a broker

Figure 2.3 presents a much more scalable approach to gateway sharing between dedicated LNSs. In this approach, only a single agent is present on each gateway, while an edge cloud broker

receives all the traffic and allows the LNSs to selectively receive relevant traffic. Such a broker usually already exists in order to interface gateways to their LNS, for example, both Chirpstack and Acklio LNS rely on the mosqitto MQTT broker for this purpose.

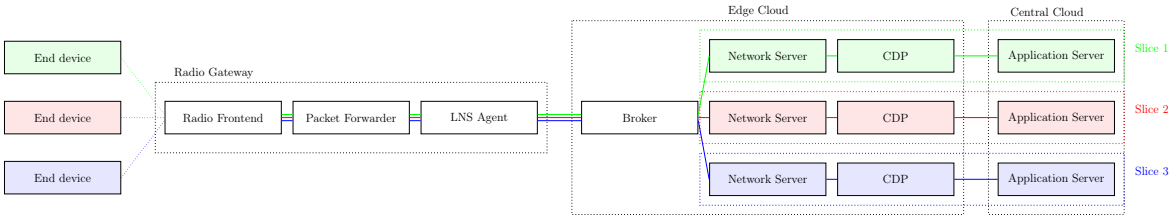

Figure 2.3: Gateway sharing through broker

Having an LNS-aware broker would however requires the gateways or the broker to be aware of the destination of a packet based on the little information available at their level (e.g. before the lorawan mac payload decoding by the LNS). A candidate identifier would be the network identifier (netid). Since this identifier is available before LNS decoding, a modified gateway agent would be able to publish uplinks using an MQTT topic derived from the netid, while LNSs would subscribe to a list of uplink topics based on the netids they want to support.

2.4 Gateway resource sharing

2.4.1 Gateway budget per tenant

Sharing gateways between several LNSs as seen above will help densify and extend LoRaWAN deployments. Gateways have however limited capacities in terms of number of messages they can process, both in terms of radio availability and in terms of duty cycle.

Figure 2.4 presents a way of addressing this issue, with gateways connected to the same infrastructure, possibly through a broker as presented in Section 2.3. Policies are introduced to connect the gateways to the tenant/slice they are attached to. Some gateways can for example be attached to one or several slices and ignore other slices. It is also possible to implement usage quotas per slice, or priority queues depending on the needs/contracts of each slice.

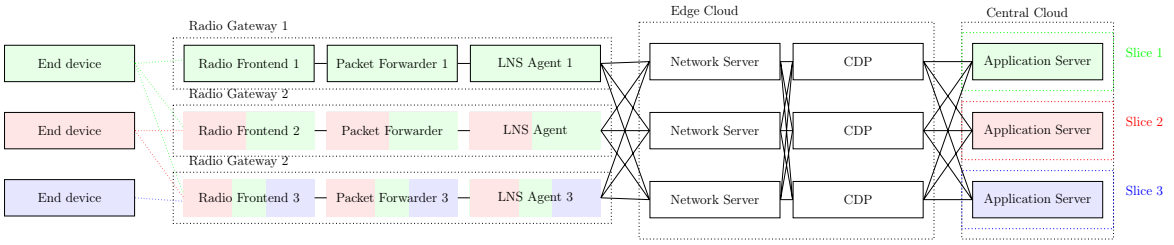

Figure 2.4: Gateway sharing a mutualized LNS infrastructure

2.4.2 Gateway spectrum assignment

Another approach to slicing would be to implement a collaboration between slices and the frequency plans of the gateways; see for instance [2, 3] and more generally WP3 results. Most LNSs have indeed ways to enable additional frequencies on gateways (through non standardized communication with their gateway agent) and on the devices (through the use of MAC commands). While such methods are usually used to improve the overall capacity of the network, one can see how this would enable a controller to dedicate frequencies, that do not overlap with others, to the devices of a given slice.

Such channel allocation would of course only isolate the slice from the rest of the devices inside the same LNS, while devices in other networks still interfering. Control mechanisms with

capabilities to detect and react to interference would therefore be needed if one was to rely on such policies for QoS.

2.4.3 Gateway directional assignment

In addition of what is described in section 2.4.2 it is also possible to ensure that a gateway will be physically able to transmit or receive a message, i.e., the radio medium has high chances of not being in use for downlink while an uplink is being received. This issue is very important when device parameter reconfiguration is needed, for instance in relation with traffic control. In order to ensure that a slice will always benefit from listening gateways, the LNSs could be configured in order to never send downlink for a fraction of the gateways. In this way, they would always be listening and able to forward a message up, as long as it reaches them.

The opposite could also be interesting. In downlink, a fraction of the gateway's downlink budget could be reserved to a given set of slices, providing them better downlink QoS or for traffic control. In this case, the gateways could also be used for uplink.

In both cases, one should ask the question of how the *set of the gateway* is selected. In particular, the amount of selected gateways should be considered in function of the required QoS. The choice of which gateways is also crucial, in particular depending on their geographic location compared to that of the slices they are expected to cover.

2.5 Security slicing for data and functional separation

An other common use case for slicing is to achieve better security by separating unrelated entities into different isolated slices. In the case of a LoRaWAN network, a LNS has to process all the frames received by its gateways to determine which belong to its devices, and needs to be able to decode at least the headers of said packets. When wanting to tightly separate two networks, the solution is often to deploy two separate LNSs. A more efficient solution would however be to exploit the two levels of LoRaWAN security (Network Session versus Application Session Keys). In this setup, each slice has its own Join Server, each capable of answering the join request of its deveui. Upon a success full join, the network server will receive and store the network session key and device address in use by the device. Making it possible to decode the device's packet's headers, but not the payloads. Attribution of the device address should help preserve anonymity of a device inside the network server (e.g. make it complicated to identify its slice), while enabling routing of its uplinks to a slice-dedicated application server. As indeed, in this setup, each slice has one or more application servers, to which were transmitted the application session key and device address that the device obtained after joining. As for the join server, these app servers are dedicated to a slice, and executed in a slice dedicated environment.

Chapter 3

Updated functional architecture

3.1 Existing architecture

The existing functional architecture of Deliverable D1.1.1 is recalled in Figure 3.1. This architecture relies on two levels of cloud platforms, namely edge and cloud platforms. Since we do not envisage in the framework of the Intelligentsia project to cloudify radio gateways, we do not consider far edge clouds, very close to devices and with objective of hosting virtual radio gateway (only some part of the middleware and of course not the antennas).

In Figure 3.1, the elements of the network impacted by the instantiation of a slice are tagged in yellow:

- The Communication Service Management Function (CSMF) and Network Slice Management Function (NSMF) functions of the orchestration platform translate the user request in terms of network service (vLNS and transport capabilities); these functions could be placed inside or outside of the orchestration platform. In the updated architecture, we shall place these functions outside of the orchestration platform.

- We have illustrated the case when a slice comprises its own Application server and portal together with a dedicated LoRa core network;

- Transport functions and controllers have to be slice aware and controllers may configure via Software Defined Network (SDN) interfaces some transport elements to reserve bandwidth or configure priority levels in the backhaul network;

- radio functions in the radio gateways as well in EDs to support slices and their SLAs.

We see that slicing impacts almost all the functions of the network, which have to be slice aware in order to take the right actions and decisions with regard to SLAs of slices.

The concept of slicing has also a major impact on the telemetry layer. The metrics coming from the radio and transport layers and the virtualized infrastructure may not be labeled with a slice identifier. Additional modules have then to be introduced in the telemetry layer in order to process data and then expose them to the orchestration layer. An additional module has to be introduced in order to make the connection between data and slices. These applies for data from the data planes as well as those from the virtualized infrastructure. This is illustrated in Figure 3.2 already present in Deliverable D1.1.1.

3.2 Updated architecture

The basic architecture remains the same but in the updated architecture, at the mid term of the project, includes the innovations developed by the various work packages of the Intelligentsia project. As in Deliverable D1.1.1, we use for slices the terminology specified by 3GPP [1].

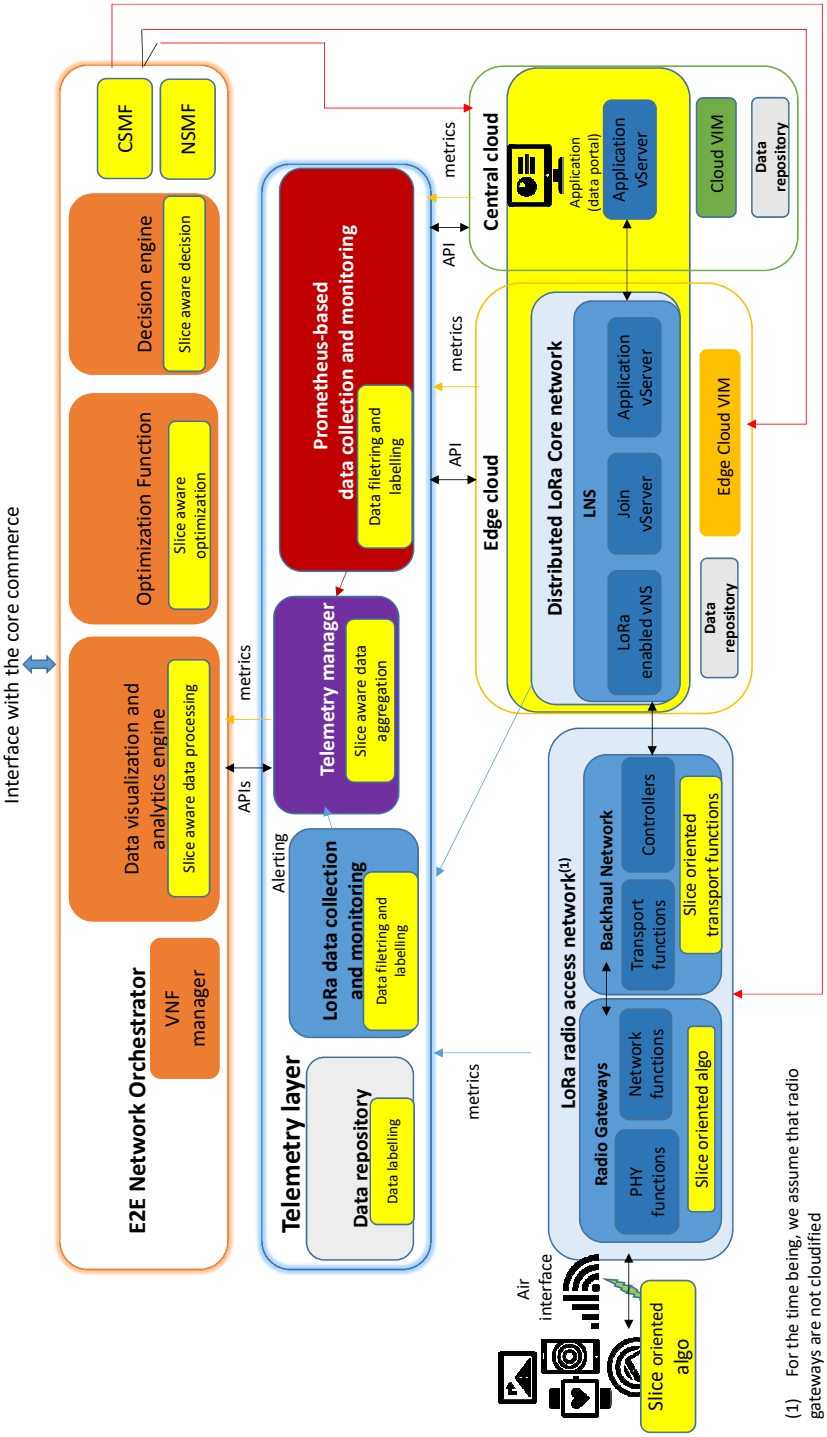

Figure 3.1: Reference functional architecture - slice aware functions are in yellow.

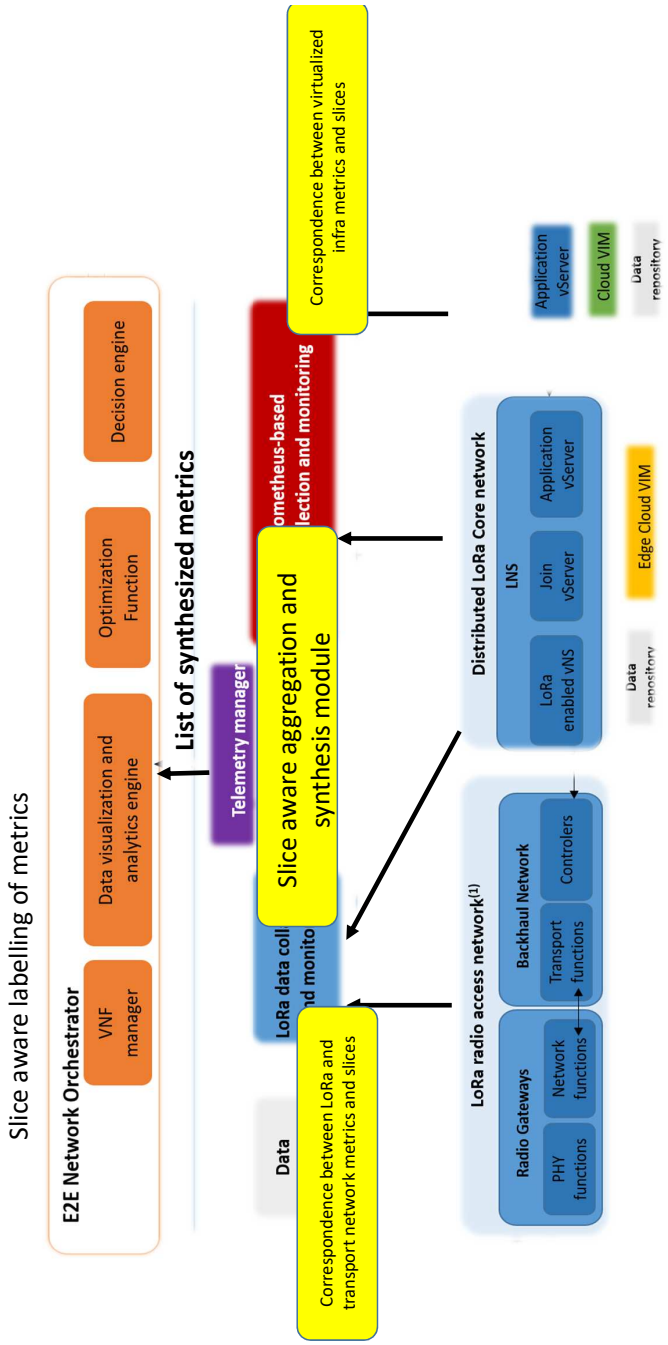

Figure 3.2: Reference functional architecture - slice aware functions are in yellow.

3.2.1 Business driven approach to slicing

Let us recall that as argued in Chapter 1, we believe that slice negotiation should be business driven and slices should be negotiated at a sufficient abstraction level. This means that slicing should not designed on the basis of technical enablers but rather on business drivers. The assurance of quality and reliability are for some companies very important drivers for developing services on top of IoT. But there are many methods of ensuring quality and reliability in networks (over-dimensioning, admission control, control mechanisms). WP3 in particular has developed several methods and algorithms to achieve quality objectives. It follows that for a same slice offer, the technical instantiation of the slice may call for various technical solutions.

In addition, from a customer to another, the way of realizing a technical slice may be different. In other words, the negotiation of a slice may involve some contextual information (real needs of a customer, the service level agreement, etc.). Hence, a direct translation of a need to a technical solution may not be adapted, which would be the case by inserting the CSMF and NSMF functions into the orchestrator, based on uniform processes. This incites us to put the CSMF and NSMF outside of the orchestrator.

The counterpart is that Monitoring and Assurance functions are crucial in case of slicing in order to meet the requirements stated in an SLA. It is then essential to introduce function for exposing network data (generated by monitoring and telemetry). Such functions exist in the 5G core network, for instance Network Exposure Function (NEF) and Network and Data Analytics Function (NWDAF). This is why we introduce a Data Exposure Function (DEF) in the functional architecture. This function exposes data computed on the basis of the results of WP2.

On the contrary, the Network Slice Subnet Management Function (NSSMF), which is in charge of the Network Slice Subnet Instance (NSSI), is left in the orchestrator platform. This choice of extracting the CSMF and NSMF from the orchestrator is discussed in more details in [5].

3.3 Multliple vLNSs

To dedicate a vLNS per slice (for instance for isolation purpose as in cellular networks where a slice can have a dedicated core), we adopt the approach described in Figure 2.2. For this purpose, we introduce the following functional elements:

- A packet forwarder function.
- LNS agent to connect to the ad-hoc Network Server.
- Network Server.
- CDP.
- Application server.

Some functions are collocated with the radio frontend (antenna and physical functions) while others are virtualized and placed on edge clouds. The Application Server (which is different of the application server of the client) can be located on a centralized cloud. The application server of the customer is not represented in the functional architecture since it is under the control of the customer.

3.4 Resource management

To take into account the algorithms developed in WP3, we introduce in the functional architecture, resource management functions, which are in charge of offering quality (e.g., PDR differentiation to slices, reliability, etc.). The resource management functions are located in both radio

gateways (LNS agent) as well in edge clouds where virtual network servers can configure radio gateways.

Resource management functions use data collected from LoRa access networks and LoRa core networks. These data are specified by WP2.

The updated functional architecture is illsutrated in Figure 3.3.

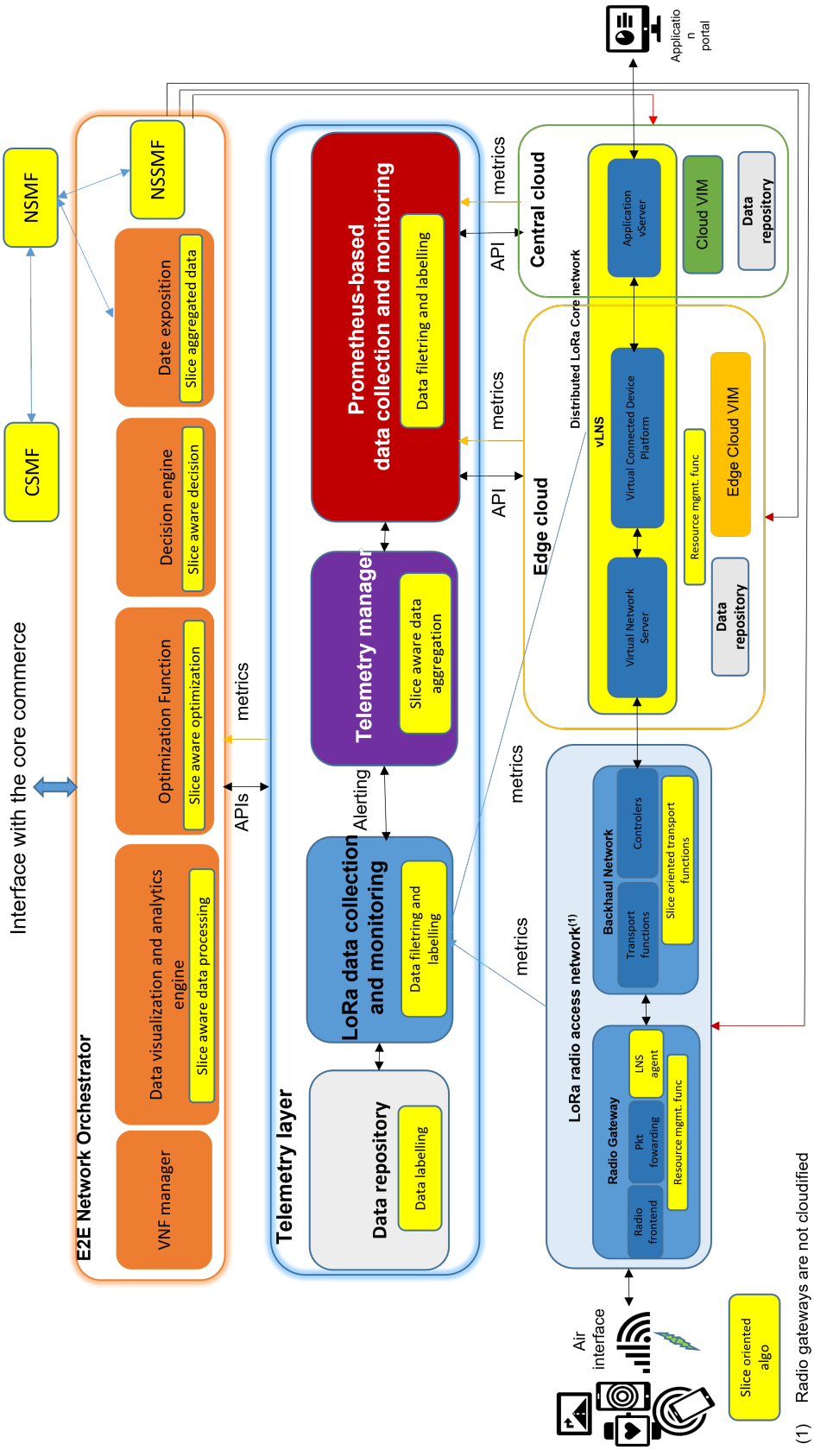

Figure 3.3: Reference functional architecture - slice aware functions are in yellow.

Chapter 4

Conclusion

We have provided in this deliverable an update of the functional architecture specified in Deliverable D1.1.1. The major differences between the two architectures are as follows:

- Introduction of resource management modules for offering differentiated quality on the air interface. These algorithms are specified in WP3 and use data collected as described in WP2.

- The virtualization of LNSs and different schemes to dedicate vLNSs to slices.

Moreover, we have advocated for a business driven approach to slicing; this motivates the separation of the NSMF function from the orchestrator.

The functional architecture specified in this deliverable will serve as a driver for setting up the testbed in WP4.

Bibliography

[1] 3GPP TR 28.801 V15.1.0. Study on management and orchestration of network slicing for next generation network (Release 15), 2018.

[2] Alessandro Aimi, Fabrice Guillemin, Stephane Rovedakis, and Stefano Secci. Packet Delivery Ratio Guarantees for Differentiated LoRaWAN Services. In *IEEE Global Communications Conference (GLOBECOM)*, Rio de Janeiro, Brazil, December 2022.

[3] Alessandro Aimi, Fabrice Guillemin, Stéphane Rovedakis, and Stefano Secci. Traffic Control and Channel Assignment for Quality Differentiation in Dense Urban LoRaWANs. In *2022 20th International Symposium on Modeling and Optimization in Mobile, Ad hoc, and Wireless Networks (WiOpt)*, Turin, Italy, September 2022.

[4] Veronica Quintuna Rodriguez and Fabrice Guillemin. Cloud-RAN modeling based on parallel processing. *IEEE Journal on Selected Areas in Communications*, 36(3):457–468, 2018.

[5] Veronica Quintuna Rodriguez, Fabrice Guillemin, and Amina Boubendir. 5g e2e network slicing management with onap. In *2020 23rd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*, pages 87–94, 2020.