A Microservices-based Approach for VNF-as-a-Service The case of IMS-as-a-Service Architecture

Amina Boubendir* †, Emmanuel Bertin‡ and Noëmie Simoni†

*Convergent Network Control Architectures Department, Orange Labs, Chatillon, France
†Networking and Computer Science Department, Télécom ParisTech, Paris, France
‡Unified Communications and Advanced Services Department, Orange Labs, Caen, France
* ‡ {amina.boubendir, emmanuel.bertin}@orange.com, †noemie.simoni@telecom-paristech.fr

***Abstract*—With the evolution of Telco systems towards 5G, new requirements emerge for delivering services. Network services are expected to be designed to allow greater flexibility. In order to cope with the new users' requirements, Telcos should rethink their complex and monolithic network architectures into more agile architectures. Adoption of NFV as well as micro-services patterns are opportunities promising such an evolution. However, to gain in flexibility, it is crucial to satisfy structural requirements for the design of VNFs as services.**

We present in this paper an approach for designing VNF-as-a-Service. With this approach, we define design requirements for the service architecture and the service logic of VNFs.

As Telcos have adopted IMS as the de facto platform for service delivery in 3G and even 4G systems, it is interesting to study its evolution for 5G towards a microservices-based architecture with an optimal design. Therefore, we consider IMS as a case of study to illustrate the proposed approach. We present new functional entities for IMS-as-a-Service through a functional decomposition of legacy network functions. We have developed and implemented IMS-as-a-Service with respect to the proposed requirements. We consider a service scenario where we focus on authentication and authorization procedures. We evaluate the involved microservices comparing to the state-of-the-art. Finally, we discuss our results and highlight the advantages of our approach.

***Index Terms*—NFV, Micro-Services, as-a-Service design, IMS, Functional Decomposition, Service Functional Components.**

I. Introduction, Motivation and Problem Outline

Within an ever-changing telecommunication ecosystem and with the evolution towards 5G systems, flexibility in delivering services is a must for Telcos. Network services are expected to be designed in a user-centric approach. Indeed, the human-centric approach will fast become ill-suited as users will range from humans (through communication services) to machines (through M2M and IoT applications). Network architectures will be transformed as Telcos adopt virtualization and softwarization at different layers of their networks. These transformations aim to achieve higher flexibility in network service delivery and higher dynamicity in network management operations.

In this direction, Network Functions Virtualization (NFV) [1] [2] promises the ability to provide agile network services. It consists of implementing network functions, that are today performed by physical dedicated hardware, in the form of Virtual Network Fnctions (VNFs) over virtualized infrastructures.

However, most of NFV-based implementations consist in cloudifying legacy network functions keep the same service logic as defined by 3GPP specifications without introducing optimization and thus reproduce the management complexity regarding the deployment and provisioning of services. Indeed, simply virtualizing existing network functions may make them cheaper to implement (COTS hardware, vendor-independency) but will not reduce network complexity or eliminate the monolithic nature of network services and architectures.

Therefore, on one hand, Telcos should reconsider their legacy functional architectures to rethink the services' logic to make them less monolithic. On the other hand, they should study optimal design for VNFs to leverage the benefits of NFV.

The optimal design is not immediately obvious, but there are resilient enablers such as Service-Oriented Architecture (SOA) [3] and Micro-Services [4] [5]. They form core elements in the technical foundation of Cloud and enterprise applications and have greatly contributed in the success of as-a-Service.

Indeed, these patterns are designed for services that need distribution over virtualized and Cloud infrastructures. This is also the case of VNFs that is why they are strong opportunities. In addition, microservices bring agile management, scalable elasticity, efficient use of resources, faster reconfiguration, a higher availability and cost-efficient redundancy [6] [7] [8].

We propose a microservices-based approach for the design of VNFs based on Atomicity, Statelessness and Loose-Coupling requirements. We apply the approach on IP Multemdia Subsystem (IMS) [9], the de facto service delivery platform for Telcos. The virtualization of IMS has been the subject of multiple works (as we present in Section II). Thus, we find it interesting to consider it as a case of study to show the evolution towards a virtualized IMS architecture with a functional decomposition and more granularity.

We survey in the following Section the related work. Then, in Section III, we present a microservice-based approach for the design of VNF-as-a-Service. Later, in Section IV, we propose a microservice-based IMS-as-a-Service architecture formalized using UML process diagrams. We focus on registration, authentication and authorization procedures. Further, in Section V, we present our experimental implementation, we describe an evaluation scenario and discuss the obtained results. This leads us to conclude our work and present the follow-up projects.

II. Related Work

To position our work, we survey in this Section, research solutions for the virtualization of IMS using NFV.

Focusing on the IMS use-case, we observe that different research works have considered the virtualization of IMS. They mainly deal with the general problem of Cloud and Telco IMS integration [10]. The authors of [11] and [12] motivate the virtualization of IMS but critically review the architectures proposed so far for cloudifying the IMS and give research directions including reconsidering granularity levels of IMS network functions.

1) A Cloud Infrastructure for IMS: The work presented in [13] proposes a cloud infrastructure for the IMS core network functions that run IMS entities on cloud-based VMs. It has proposed a resource allocation algorithm to scale IMS vertically and adapt to loads. However, VNFs are exactly the legacy IMS functional entities (P-CSCF, S-CSCF, HSS,...) keeping the same service logic as defined today by 3GPP specifications. However, this means keeping the stateful and highly coupled architecture of IMS functional entities. This does not allow horizontal scalability. It also proposes a resource allocation algorithm for high resource utilization. But, resource efficiency may not be maximal as functional decomposition into more granular entities has not been considered.

The work in [14] presents three IMS-as-a-Service architectural models for virtualized IMS based on NFV as follows:

2) Virtualized IMS: Virtualized-IMS architecture [14], like in [13], has implemented each legacy IMS entity as software on a single VM keeping the same interfaces between entities. It can somehow scale using the procedures standardized by 3GPP. However, the reproduction of the same functional service logic of IMS stateful architecture limits the scalability.

3) Split IMS: The Split-IMS proposes a general architecture where IMS entities are split into multiple NFWorkers to be deployed on top of multiple VMs or containers. It also moves the state that these NFWorkers maintain to an external functional entity called Shared-Memory to make them stateless. However, even though it considers state separation and functional split, it does not propose a fine-grained decomposed architecture describing roles for NFWorkers. This is important to verify the optimality of the proposed split and how it affects the scalability, performance and resource efficiency.

4) Merge IMS: The Merge-IMS deploys the four main IMS entities (P-CSCF, S- CSCF, I-CSCF, and HSS) into one IMS-VM. A new IMS-Locator entity assigns the subscribers to a dedicated IMS-VM at registration. HSS entities of each IMS-VM share a same database (SharedDB). Scaling is possible by instantiating a new IMS-VM with all entities. Thus, scaling is difficult due to the monolithic and stateful nature of IMS-VM.

Existing approaches perform re-engineering and do not address optimal decomposition of functional entities. Also, they do not consider microservices for the software design. This is an obstacle for agile management and elastic scalability. Therefore, we propose next, an approach to re-architect network functions into granular and stateless microservices.

III. A micro-service approach for VNF-as-a-Service

In this Section, we present our approach for the design of VNFs as microservices. We describe the approach through a set of requirements for the service logic design.

The microservice architectural model is an approach to develop a single application as a set of elementary services [4]. This copes with our objective of decomposing Telco network functions into sub-entities with finer granularity levels to eliminate their monolithic nature. We propose the following micorservice-based approach for an efficient decomposition.

1) Separation of State: The first step in reconsidering the granularity levels of a network service or architecture is to consider separating the service logic of functional entities and the handled data or state. This is to build **stateless services**. A service is stateless if it processes each request as an independent transaction without any relationship with previous ones [15]. It should then neither keep information regarding its state or its processing state nor handle information about previous requests. Each service should handle data coming only from outside its area of responsibility. Information handling and state management should be delegated to an external data base. With the existing approaches, it is impossible to terminate a VNF instance when it handles ongoing calls as this requires transferring the stored state to another VNF instance. Therefore, with stateless entities the logic of a service can scale easily by instantiating new stateless entities.

2) Functional Decomposition: The next step is to consider the granularity level of functional entities by decomposing the logic into elementary **atomic services**. It is important to list all the functions that an entity is responsible of and extract them to create new ones. The new functional entities should perform one and only one service. However, it should be cohesive and consistent with a relevant service logic recognized as a meaningful business service for potential users (business models may determine granularity). It should be autonomous to find all its functionalities in a logical way internally.

3) Functional Independence: The last step consists in defining the new service processes while verifying that services are **functionally independent or loosely coupled**. This means that services should have no predefined sequence between them. At run-time, they should maintain relationship with minimized functional dependencies. The loose coupling allows to verify the stateless and the atomicity requirements. Indeed, if a service maintains its state for a long period, it will lose the loose coupling requirement and thus its availability for other incoming requests and even its possibility to scale. Also if a service needs to rely on another service, then it is tightly coupled to it. For services composed of a set of elementary services, there should be reduced functional dependencies among interacting service components. These requirements are a must for building agile and scalable distributes services.

This approach provides guidelines for the functional decomposition and the design of VNFs as microservices. We rely on this approach to study the functional decomposition of IMS network functions as we present in the next Section.

IV. IMS-as-a-Service Use Case
A microservice-based IMS architecture

In this Section, we present our proposition of IMS-as-a-Service architecture using the proposed VNF-as-a-Service design approach. It is the result of a functional decomposition of legacy IMS network functions [9] where we have reconsidered granularity levels of IMS network functions into new smaller, elementary sub-functional entities according to the proposed design requirements. Thus, this has involved separating the functional logic of entities from data and state.

In order to describe the proposed functional architecture, we formalize elementary functions and functional data. According to best practices in service architecture and software engineering, we suggest using stereotypical UML categories. We formalize the functional blocks using Service Functional Component (SFC) UML category where **each SFC is a micro-service**. The data that these functional blocks handle is formalized as Service Functional Data (SFD).

A. Description of IMSaaS Functional Architecture

Figure 1 presents a functional architecture for IMS-as-a-Service. It describes the new functional blocks and the data necessary for a registration procedure of a UE (terminal) to the network. The procedure of terminal registration consists in the establishment of a "temporary" link between an equipment connected to the network (under network coverage with an IP address) and a reachability address, such as a telephone number. The proposed functional view describes the execution of a network service scenario and its stakeholders via a sequence of functional operations. The execution of the service in the sequence is seamless. It does not take into account technical boundaries (the used terminal, platform, or network) neither responsibility boundaries (elementary services in the case of a composed service). This functional view thus corresponds to an end-to-end vision of the service.

1) Registration Procedure: The actors involved in the terminal registration procedure are the end user, the network provider and the service provider. Please note that the network provider and service provider may be the same actor.

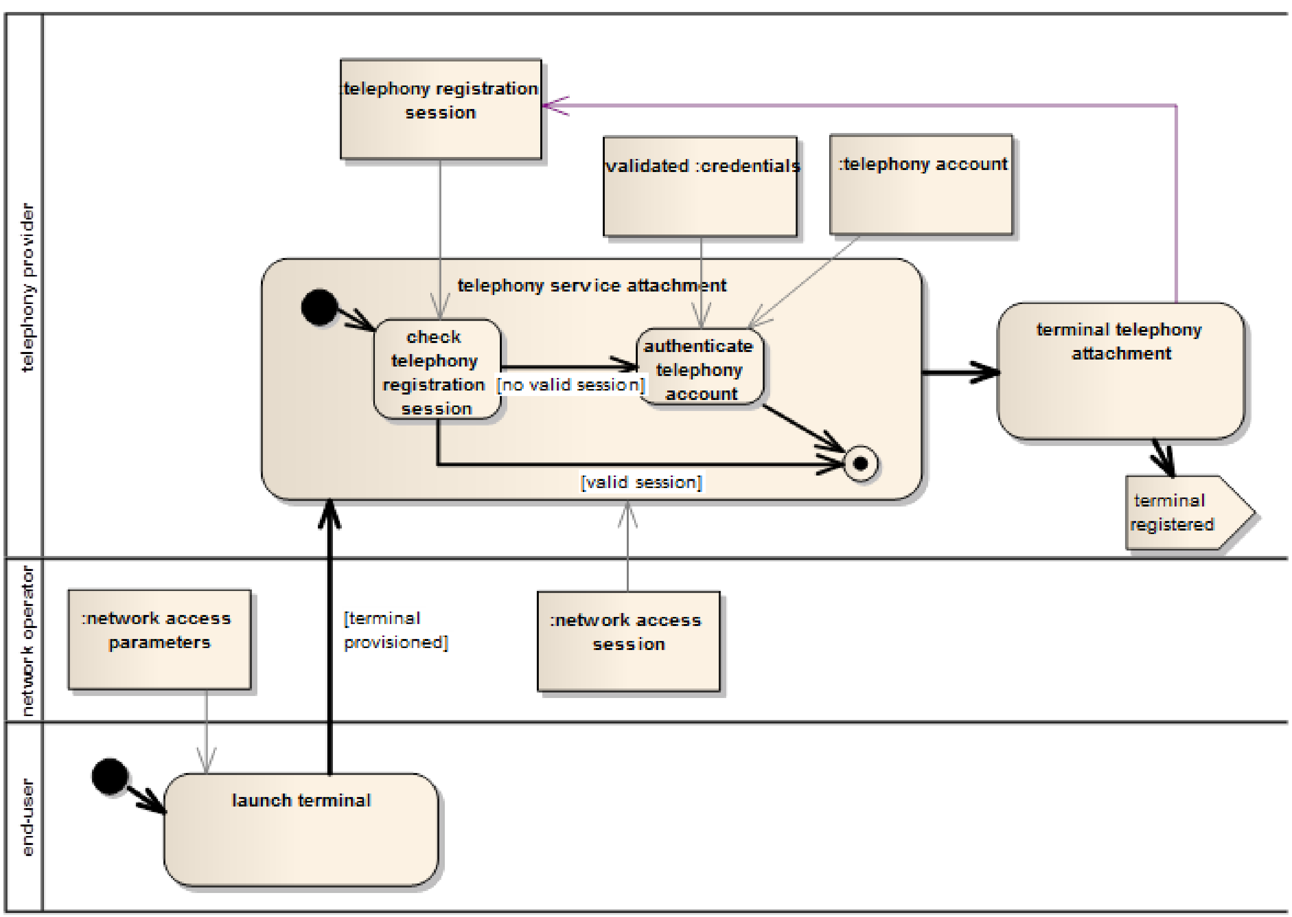

Fig. 1: IMS-as-a-Service: microservices-based vIMS Registration procedure

The SFCs involved in the registration procedure are:

- *Launch terminal* SFC: it is used to initialize the terminal registration request. It uses the *network access parameters* SFD provided by the network. Typically this is an IP address as it is connected to the network.
- *Service attachment* SFC: receives the registration request and is responsible for establishing the attachment of the terminal to the service. This is a two-operation SFC, which makes it a composite SFC, composed of *check registration session* and *authenticate account* SFCs.
  – The *check registration session* verifies if the user/terminal has an already registration session. For that, it checks the *registration session* SFD (such as a SIP Address Of Record) required for registration into HSS. If a session is already running and valid, then the *service attachment* retrieves *network access session* SFD and its process ends.
  – If there is no valid session, then there is a need for authentication. The *authenticate account* corresponds to the functional operation that checks access right to the network. It uses *credentials and account* SFDs (SIP login, password, IMPI, authentication keys,...) to authenticate the user. For that, it interrogates the HSS as it holds the initial account subscription. We detail the description of the authentication procedure further in this Section.
- *Terminal attachment* SFC: is responsible for the attachment of the terminal to end the registration procedure. It uses some SFDs and test results from the previous SFCs to finally register the terminal.

We have chosen to focus on authentication and authorization procedures: on *authenticate account* composite SFC as it involves other elementary SFCs), and on authorization procedure through a scenario of making a call as we describe here after.

2) Authentication Procedure: Authentication relies in verifying the user identification, if the user is known by the network provider and service provider and if he has the right to access the network or service Figure 2presents the functional building blocks of *authenticate account* SFC. The SFCs involved in the authentication procedure are:

- *Check account* SFC: checks if the user is known and has a subscription (a user account and profile). For that it checks the *account* SFD within the HSS.
- *Select authentication means* SFC: is responsible for gathering SFDs required for the authentication and that are provided by all involved actors. That is: the user-provided credentials (login and password), the network access session and the Customer Premises Equipment (CPE) credentials as well as account SFDs.
- *Check credentials* SFC: is responsible for the verification of all the provided credentials by interrogating HSS.

After authentication, the *authenticate account* SFC process provides registration session SFD.

3) Authorization Procedure: Figure 3 describes the service process of making a call. It involves authentication and authorization. At subscription, the service provider determines the profile and rights attached to a user's account stored in a data-base (HSS). Authorization procedure is performed when a user requests to access a service. It is performed only if the user has been authenticated by the service provider via a user account and credentials or through its terminal (e.g. a SIM card). Authorization is performed by *authorize call* SFC (see Figure 3). It is responsible of checking whether the user is requesting access to a subscribed service. Thus, it checks and controls the user rights to access a target service or a content according to its subscription profile by interrogating HSS to retrieve *requester account* SFD.

The actors involved are the caller, the callee and the service provider. When a consumer of the service wishes to communicate vocally and instantaneously with a contact, the caller user selects the telephone number and then makes the call, provided that he/she has the right to access the service. The service provider delivers the call, after having verified the validity of the callee's telephone number. If the callee answers, this produces a communication session that allows the caller and the callee to communicate. The verification of rights (authorization) is carried out at the time the call is requested.

B. Compliance with VNF-as-a-Service Requirements

The SFCs that build our proposition of IMS-as-a-Service are compliant with the requirements defined by the proposed microservice-based VNFaaS approach. Indeed, as described above, SFCs are stateless because when performing the service, they retrieve the SDFs that they need to handle (data and state) externally, mainly from a centralized data base equivalent to HSS where these SDFs are stored. HSS should be re-designed as a more resilient data base as presented in [16]. They also rely on another external entity that keeps state as we describe in the next Section. Also, each SFC is responsible of one (elementary) service. They are also involved in composition (authentication example) where each service of the composition is independent from the other services.

We summarize in Table I below, the compliance of existing vIMS architectures with our VNFaaS approach.

	Seperation of state	Functional decomposition	Functional independency
Cloud for IMS [13]	x	x	x
Virtualized IMS [14]	x	x	x
Split IMS [14]	✓	x	x
Merge IMS [14]	x	x	x
IMS-aaS approach	✓	✓	✓

TABLE I: Compliance of vIMS with VNFaaS approach

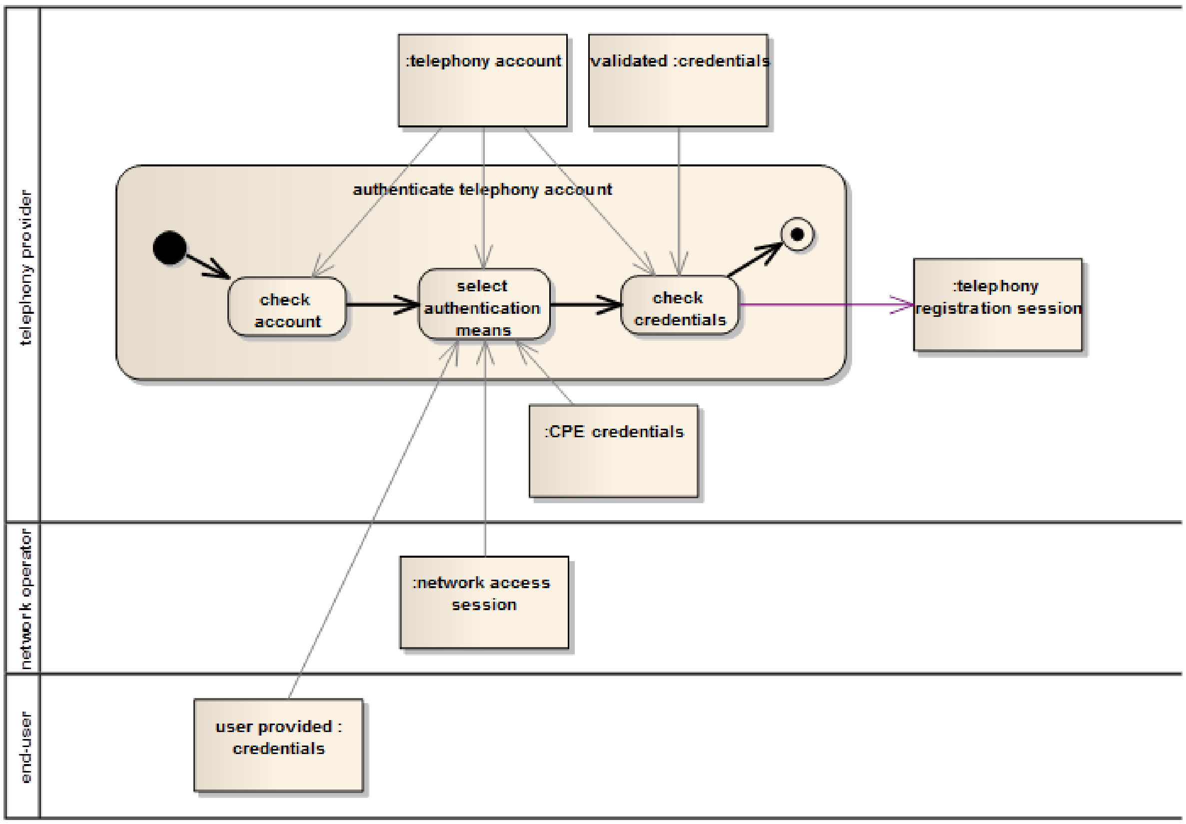

Fig. 2: Authentication procedure in IMS-as-a-Service

V. Implementation and Evaluation Results

We present in this Section details about the implementation of the proposed microservice-based IMS architecture, the conducted experiments and the obtained evaluation results.

A. Implementation Environment

We have developed the proposed microservices by analyzing open source OpenIMS solution [17]. We have then developed the new microservices functional entities using VerCors [18], a design platform for the modeling, specification, verification and validation of the software architecture of services. After validation, the services are then instantiated and executed using GCM/Proactive [19] which ensures the life-cycle management of microservices (SFCs).

Then, as the developed services are stateless, we have developed an external entity called *Frontal* to handle state. Indeed, the SIP protocol [20] is a low level signaling protocol that needs expert knowledge. Current developments of IMS or VoIP services are usually based on the use of a SIP stack which implements the behavior of a SIP UA. Therefore, there is a need to hide the SIP complexity and overall to keep state outside SFCs. This makes SFCs protocol-independent. In addition, the interaction between microservices should no longer use IMS 3GPP interfaces and protocols.

Thus, *Frontal* is a protocol automate and state automate. It is responsible of converting SIP signaling messages (Register, Invite,...) into Java RMIs (Remote Method Invocation), lightweight reliable and scalable interfaces. It is also responsible of keeping state (SFDs) as an intermediate between end-user terminal and the IMS-aaS architecture.

B. Experiments and Evaluation

1) Experimental scenario: In order to evaluate the proposed IMS-aaS microservices, we have performed the scenario of Figure 3 which involves the process of Figure 2. We have simulated thousands of simultaneous SIP calls using a VoIP call generator, StarTrinity SIP Tester [21].

The call generator acts as end users. It sends SIP requests towards the *Frontal* external (stateful) entity. *Frontal* converts the SIP into Java RMI and relays them to the first SFC of the process. During the process, SFDs are retrieved from an experimental DB.

2) Evaluation metrics: We measure:

- The response time (or processing time) of the two procedures for an increasing number of requests.
- The resource efficiency: the consumption of CPU and memory by the procedures.

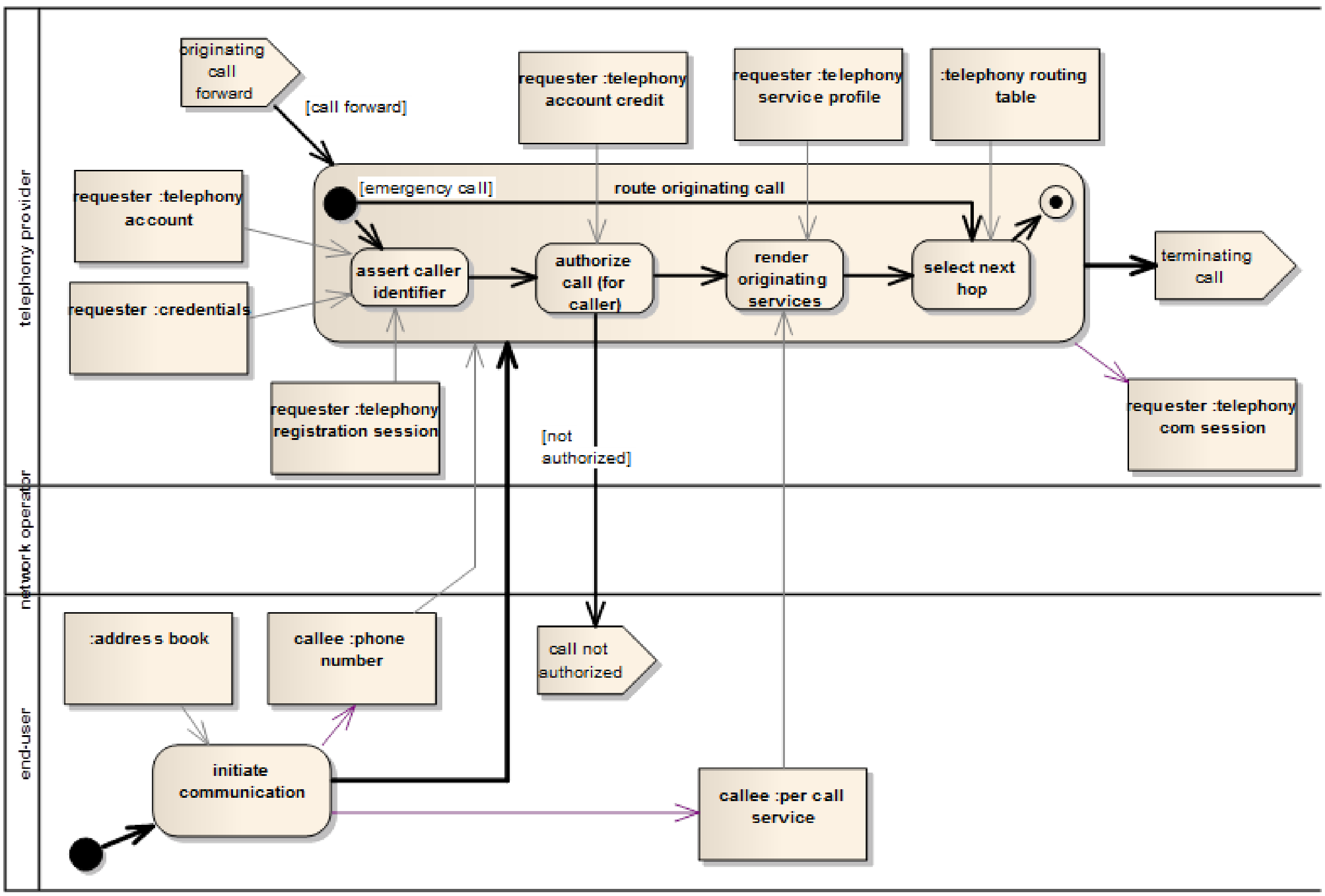

Fig. 3: Call authorization procedure in IMS-as-a-Service

C. Evaluation results

Based on the evaluation scenario and metrics descried above, we present here the evaluation results regarding the response time for the authentication and the authorization procedures. We also present the response time of the DB since these two procedures involve invocating it. Figures 4, 5 and 6 represent respectively the response time of authentication procedure, authorization procedure and DB invocation. The X-axis represents the number of simultaneous requests.

Figures 7 and 8 and Figures 9 and 10 show respectively the results regarding the CPU usage and the memory usage by both authentication and authorization processes including all involved microservices.

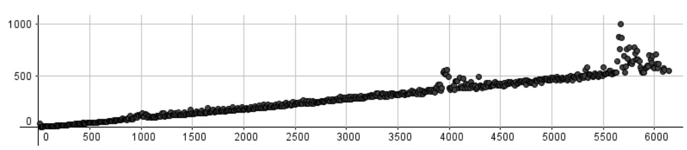

Fig. 4: Response time of Authentication procedure

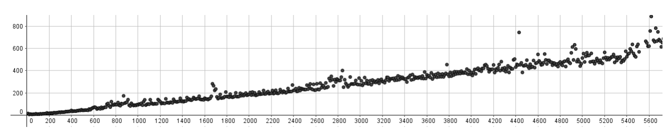

Fig. 5: Response time of Authorization procedure

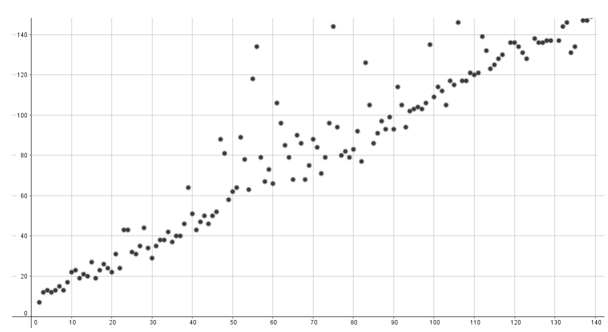

Fig. 6: Response time of DB (HSS) invocation

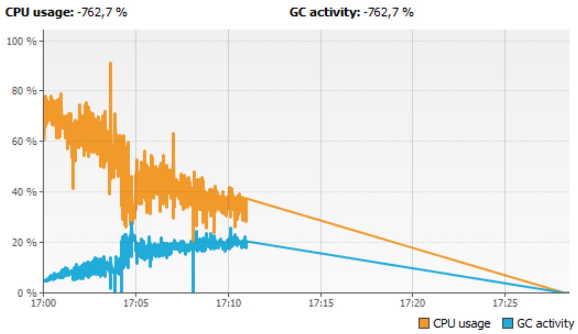

Fig. 7: CPU usage by authentication procedure

D. Discussions

According to our evaluation results on the response or processing time of the IMS-as-a-Service microservices, we can say that they can handle a large number of subscribers requests. Indeed, authentication process responds to a maximum of 5500 simultaneous authentication requests in less than 500ms. Also, authorization process responds to 5600 simultaneous authorization requests within less than 600 ms. These response times include the processing time of all involved microservices and the time needed to retrieve SFDs from the database.

Regarding resource consumption, comparing to evaluation results from Split-IMS [14] which also considers statless network functions, our evaluation results present better resource efficiency results on CPU usage in % and on memory usage.

Furthermore, the design of VNFs using the microservices approach with optimal granularity level sets the foundation for addressing elastic scalability issues and research questions regarding life-cycle automation, on-demand instantiation and exposition of VNF-as-a-Service through a Telco PaaS. Indeed, a Telco PaaS would be responsible of exposing VNFs through APIs and automating the life-cycle management (deployment, monitoring, auto-scaling, auto-recovery and orchestration). For Telcos, such a PaaS would run on distributed infrastructures interconnected through the WAN in order to ensure service continuity and reduced latency.

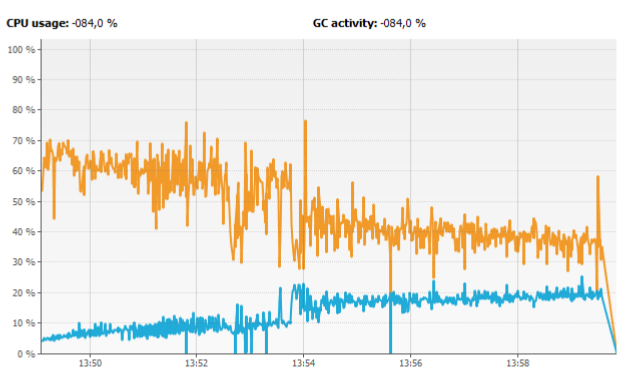

Fig. 8: CPU usage by authorization procedure

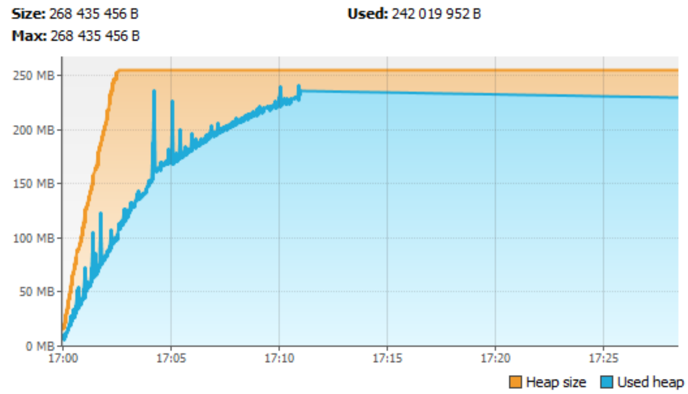

Fig. 9: RAM usage by authentication procedure

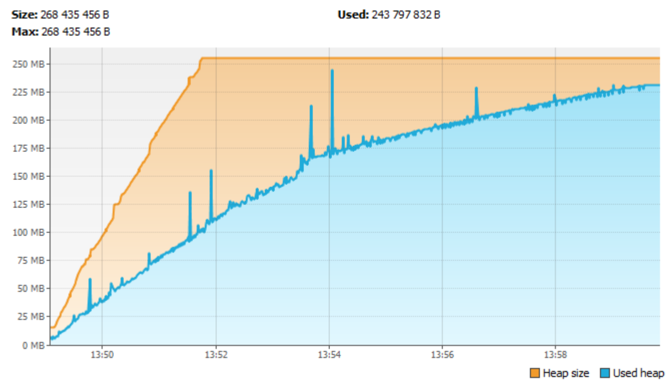

Fig. 10: RAM usage by authorization procedure

VI. Conclusions & Future Work

We have presented in this paper a mircoservice-based approach for the design of VNFs as services. The use of micorservices software pattern for VNFs is motivated by a move towards more flexible service delivery and more agile network management operations.

We have studied the impact of the design approach onto IMS, a widely studied illustration case. We have shown the compliance of our approach with the design requirements of microservices and the advantages on QoS parameters such as availability through the measurement of response time, and the resource consumption efficiency comparing to existing approaches. As a next step, we will study modes of exposition of VNF-as-a-Service in a Telco PaaS mode for on-demand deployments.

Acknowledgment

We would like to thank Mr. Frédéric Lemoine from CNAM (Conservatoire National des Arts et Métiers) and AIRS research team (Architectures et Integration des Réseaux et des Services) from Télécom ParisTech for the valuable help to achieve this work. We also would like to thank Mr. Prosper Chemouil for his helpful remarks and suggestions upon writing the paper.

References

[1] B. Han, V. Gopalakrishnan, L. Ji, and S. Lee, "Network function virtualization: Challenges and opportunities for innovations," *Communications Magazine, IEEE*, vol. 53, pp. 90–97, Feb 2015.

[2] *ETSI NFV ISG, Network Functions Virtualisation (NFV); Architectural Framework. Group Specification*, Dec 2014.

[3] A. V. V. Q. Duan, Y. Yan, "A survey on service-oriented network virtualization toward convergence of networking and cloud computing," *IEEE Transactions on Network and Service Management*, vol. 9, pp. 373–392, dec 2012.

[4] "Microservices. a definition of this new architectural term. http://martinfowler.com/articles/microservices.html," 2016.

[5] D. Namiot and M. Sneps-Sneppe, "On micro-services architecture," *International Journal of Open Information Technologies*, vol. 2, no. 9, 2014.

[6] D. Malavalli and S. Sathappan, "Scalable microservice based architecture for enabling dmtf profiles," in *Network and Service Management (CNSM), 2015 11th International Conference on*, pp. 428–432, Nov 2015.

[7] M. Villamizar, O. Garcs, H. Castro, M. Verano, L. Salamanca, R. Casallas, and S. Gil, "Evaluating the monolithic and the microservice architecture pattern to deploy web applications in the cloud," in *Computing Colombian Conference (10CCC), 2015 10th*, pp. 583–590, Sept 2015.

[8] D. Jaramillo, D. V. Nguyen, and R. Smart, "Leveraging microservices architecture by using docker technology," in *SoutheastCon 2016*, pp. 1–5, March 2016.

[9] 3GPP, "Ip multimedia subsystem (ims), stage 2, 3gpp is 23.228, vol. 12," 2015.

[10] F. Gouveia, S. Wahle, N. Blum, and T. Magedanz, "Cloud computing and epc / ims integration: New value-added services on demand," in *Proceedings of the 5th International ICST Mobile Multimedia Communications Conference*, Mobimedia '09, pp. 51:1–51:5, 2009.

[11] R. Glitho, "Cloudifying the 3gpp ip multimedia subsystem: Why and how?," in *New Technologies, Mobility and Security, 2014 6th International Conference on*, pp. 1–5, March 2014.

[12] M. Abu-Lebdeh, J. Sahoo, R. Glitho, and C. W. Tchouati, "Cloudifying the 3gpp ip multimedia subsystem for 4g and beyond: A survey," *IEEE Communications Magazine*, vol. 54, pp. 91–97, January 2016.

[13] F. Lu, H. Pan, X. Lei, X. Liao, and H. Jin, "A virtualization-based cloud infrastructure for ims core network," in *Cloud Computing Technology and Science, 2013 IEEE 5th International Conference on*, vol. 1, pp. 25–32, IEEE, 2013.

[14] G. Carella, M. Corici, P. Crosta, P. Comi, T. M. Bohnert, A. A. Corici, D. Vingarzan, and T. Magedanz, "Cloudified ip multimedia subsystem for network function virtualization (nfv)-based architectures," in *Computers and Communication, 2014 IEEE Symposium on*, vol. Workshops, pp. 1–6, June 2014.

[15] M. Kablan, B. Caldwell, R. Han, H. Jamjoom, and E. Keller, "Stateless network functions," in *Proceedings of the 2015 ACM SIGCOMM Workshop on Hot Topics in Middleboxes and Network Function Virtualization*, HotMiddlebox '15, (New York, NY, USA), pp. 49–54, ACM, 2015.

[16] T. Yang, X. Wen, Y. Sun, Z. Zhao, and Y. Wang, "A new architecture of hss based on cloud computing," in *Communication Technology (ICCT), 2011 IEEE 13th International Conference on*, pp. 526–530, Sept 2011.

[17] Fraunhofer, "Fokus openimscore, www.openimscore.org," tech. rep.

[18] "Verification of models for distributed communicating componants. https://team.inria.fr/scale/software/vercors/vcev4-download/."

[19] "Proactive parallel suite. http://proactive.inria.fr/."

[20] "Session initiation protocol. http://www.rfc-base.org/txt/rfc-3261.txt," tech. rep.

[21] StarTrinity, "Sip tester, call generator and simulator. http://startrinity.com/voip/siptester/siptester.aspx," 2016.