New architecture and function to improve autonomy, dynamicity, and intelligence of future network management

Guy Saadon

▶ **To cite this version:**

Guy Saadon. New architecture and function to improve autonomy, dynamicity, and intelligence of future network management. Networking and Internet Architecture [cs.NI]. Institut Polytechnique de Paris, 2022. English. NNT : 2022IPPAT042 . tel-03910691

HAL Id: tel-03910691
https://theses.hal.science/tel-03910691

Submitted on 22 Dec 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INSTITUT POLYTECHNIQUE DE PARIS

TELECOM Paris

IP PARIS

NNT : 2022IPPAT042

Thèse de doctorat

New architecture and function to improve autonomy, dynamicity, and intelligence of future network management

Thèse de doctorat de l'Institut Polytechnique de Paris
préparée à Telecom-Paris

École doctorale n°626 Ecole Doctorale de l'Institut Polytechnique de Paris (ED IP Paris)
Spécialité de doctorat : Informatique

Thèse présentée et soutenue à Massy-Palaiseau, le 23/11/2022, par

Guy Saadon

Composition du Jury :

M. Samir Tohme Professeur, Université de Versailles Saint-Quentin, France	Président
Mme Gordana Gardašević Professeur, Université de Banja Luka, Bosnie, Herzégovine	Rapporteur
M. Kurt Tutschku Professeur, Blekinge Institut de Technologie, Karlskrona, Suède	Rapporteur
M. Claude Chaudet Professeur, Webster, Université de Genève, Suisse	Examinateur
M. Tobias Hossfeld Professeur, Université de Würzburg, Allemagne	Examinateur
M. Jose Soler Professeur, DTU fotonik, Danemark	Examinateur
M. Gerard Memmi Professeur, IPP, Telecom ParisTech	Directeur de thèse
M. Yoram Haddad Professeur, Jérusalem Collège of Technologie, Israël	Co-Directeur de thèse
Mme Noémie Simoni Professeur, IPP, Télécom ParisTech	Invité

Dedication

This thesis is dedicated to my dear wife Arièle for her understanding, her patience, and for her support for a project that in the eyes of many, may seem a little unreasonable.

Acknowledgments

This research was partly funded by the Israel Innovations Authority under the Neptune generic research project. Neptune is the Israeli consortium for network programming. I would like to thank all the operators and vendors of this consortium who helped us to illustrate our articles with realistic examples and to keep it close to the reality of the industry.

I would like to express my deep gratitude to:

Prof. Yoram Haddad, for initiating the project and organizing the scientific and financial support for this PhD. I would also like to thank him in particular, for his kindness, continuous support, his demand for excellence, and infinite patience.

Prof. Noëmie Simoni, for her support in Telecom Paris / University Paris-Saclay / IPP, for her delicacy, her intelligence, her kindness, and not less than Yoram: infinite patience.

Dr. Michael Dreyfuss for his effective help for the third article, and MM. Doron Solomon, ex-CTO of ASOCS, for his kind help and listening to better define the use cases and the relevance of the defined rules.

Pr. Gérard Memmi for having welcomed me into his department, for following the progress of my work and for having supported and trusted me during these 6 years.

Chana Klein and Florence Besnard, for JCT and IPP, for their support to my PhD and especially in the complex administrative tasks.

Pr. Dan Bouhnik, Dr. Philippe Ezran, Dr. Azriel Heuman, and Aryeh Wiesen. They are always ready to help at the Jerusalem College of Technology.

Pr. Talel Abdessalem, for accepting my application and allowing me to carry out this thesis.

Dr. Nissim Amzalleg for his friendship, advises, and support during my thesis.

My children and my family, and the way they have accepted my career change and lack of availability. They recognized that realizing a project/dream and doing everything to realize it, is always worth it even if staying in high-tech could have seemed more comfortable. I hope this will remain an example in their lives, a life often being paved with pitfalls.

The people of Israel. They are so full of life and resources that when we see them live and bounce back, anything seems possible.

And God, of course, who is always here even if it is not always easy to understand the way he manages his world.

« La vraie vie est si souvent celle qu'on ne vit pas » Oscar Wilde

« Alors donnons-nous une chance de la vivre même si ce n'est pas toujours la voie la plus simple. »

List of Figures

List of tables

Table of contents

Abstract

Résumé de thèse :

Avec le déploiement actuel des réseaux 5G et IoT, nous assistons à une explosion de la demande dans le domaine des mobiles et des applications. Les réseaux deviennent plus complexes et difficiles à gérer. L'automatisation reste limitée et les interventions humaines peuvent générer des erreurs. Une analyse de l'industrie et de la recherche révèle un besoin croissant d'autonomie et de dynamicité. La complexité des demandes et leur nombre requièrent aussi plus d'intelligence.

Dans ce contexte, comment pouvons-nous participer à l'évolution des réseaux afin d'assurer cette autonomie et dynamicité ? les actuelles technologies et architectures sont-elles suffisantes ? Riche d'une expérience de 20 ans dans l'industrie des télécoms, ces questions m'ont interpellées. Aussi, cette thèse propose des réponses, à l'aide d'une nouvelle architecture et de l'introduction d'intelligence dans la gestion de réseaux.

Du côté de la standardisation, l'orchestration ajoute de l'intelligence et une plus grande autonomie. Les contrôleurs SDN programmables et la virtualisation du réseau assurent une dynamicité partielle. Côté recherche, l'utilisateur est au centre et utilise des composants virtuels et dynamiques afin de créer lui-même ses propres services. Dans le domaine technologique, le découpage du réseau, le cloud, les micro-services, et l'intelligence artificielle (IA) s'imposent.

Cependant, la dynamicité centrée sur l'utilisateur, en particulier si sa demande évolue en cours de session, n'est pas vraiment assurée. L'orchestration reste monolithique et de niveau ressources réseau. Elle doit répondre aux besoins des différentes couches de la gestion mais ne permet pas une réelle autonomie de chaque couche. L'autonomie reste limitée et le réseau, même muni d'un orchestrateur ne peut répondre à tous les scenarios. Ainsi nos contributions se concentrent sur ces trois points : autonomie, dynamicité, et intelligence.

Notre première proposition est d'ordre architecturale et organisationnelle. Une couche de réseau virtuelle est introduite et permet une virtualisation des services de bout en bout avant l'allocation physique des ressources. Cette approche procure une plus grande dynamicité dans la gestion des services par l'utilisateur. L'orchestration est distribuée sur 5 couches à savoir, utilisateur, services et applications, slices et services virtuels, ressources réseaux, et technologies, afin de garantir une plus grande autonomie et de meilleures performances.

Notre seconde proposition est d'ordre fonctionnelle. L'orchestrateur de slices et services virtuels a pour rôle le déploiement virtuel et le contrôle des services demandés et de leurs composants VNFs en intégrant les contraintes QoS locales et de bout de bout. A l'aide d'une simulation, nous

montrons l'intérêt de cette architecture afin de démontrer la dynamicité de la demande utilisateur même en cas de changement des paramètres de services durant une session.

Notre troisième proposition a trait à l'autonomie et à l'introduction d'intelligence à l'intérieur de l'orchestrateur de ressources. Lors de conflit de ressources, celui-ci dispose de règles permettant d'améliorer le ratio de services allouables. Plusieurs règles peuvent être applicables mais celles-ci peuvent entrer en conflit les unes avec les autres. Nous proposons donc une fonction d'aide à la décision, capable de choisir la meilleure règle à appliquer, basée sur l'évaluation de chaque règle. Cette fonction tient compte de nombreux paramètres tels que le coût, le profit, ou le respect du contrat de service. Une analyse numérique démontre son intérêt et sa faisabilité.

Ces trois propositions participent donc à l'amélioration de l'autonomie, de la dynamicité, et de l'intelligence des gestionnaires de réseaux. L'orchestration encore à ses débuts et L'IA pénètrent peu à peu les réseaux et les micro-services participent à la dynamicité. Notre recherche axée sur l'amélioration des solutions existantes, vise le graal du « zero-touch ».

Résumé de thèse pour le grand public :

Les réseaux 5G et IoT face à une explosion de la demande, deviennent plus complexes et difficiles à gérer. L'automatisation est limitée et les interventions humaines génèrent des erreurs. Côté forums, l'orchestration, les contrôleurs SDN, et la virtualisation réseau introduisent une dynamicité partielle. Côté recherche, la personnalisation impose agilité et intelligence. Cependant, l'orchestration est monolithique. La dynamicité et l'autonomie relatives au « on-demand » ne sont pas assurées. Aussi, face à ces nouveaux défis, nos contributions tentent de répondre à ces besoins. Notre première proposition, architecturale et organisationnelle introduit une couche pour concevoir et gérer les réseaux virtuels. Notre orchestration est distribuée sur 5 couches afin de garantir autonomie et performances. Notre seconde proposition, fonctionnelle, supportée par une simulation, adresse la dynamicité des services « on-demand ». Notre dernière proposition soutenue par une analyse numérique, est une fonction d'aide à la décision, basée sur le SLA de haut niveau afin d'améliorer le ratio de services allouables. Face à ces besoins d'autonomie, de dynamicité, et d'intelligence du nouvel écosystème, notre recherche vise le graal du « zero-touch ».

Abstract:

With the rise of 5G and IoT networks, we are witnessing an explosion of demand for mobiles and applications. Networks are becoming increasingly complex and difficult to manage. Even if part of the operations is automated, human intervention is still necessary on a daily basis, which can lead to errors. An analysis of industry and research reveals a growing need for autonomy and dynamicity. The complexity of the requests and their number also require more intelligence.

In this context, how can we help transform networks to ensure this autonomy and dynamicity? Are current technologies and architectures sufficient? With 20 years of experience in the telecom industry, these questions challenged me. Therefore, this thesis tries to provide solutions, using a new architecture and the introduction of intelligence in network management.

On the standardization side, orchestration adds intelligence and greater autonomy. Programmable SDN controllers and network virtualization ensure partial dynamicity. On the research side, the user is at the center and uses virtual and dynamic components to create his own services. In technology, network slicing, cloud, micro-services, and artificial intelligence (AI) are needed to meet these needs.

However, the user-centric dynamicity, in particular if his service request evolves during the session, is not really assured. Orchestration remains monolithic and at the network resource level. It must meet the needs of the different layers of management but does not allow the autonomy of each layer. Autonomy remains limited and the network, even equipped with an orchestrator, is not able to respond to all scenarios. So, our contributions focus on these three points: autonomy, dynamicity, and intelligence.

Our first proposal is architectural and organizational. A virtual network layer is introduced and allows end-to-end service virtualization before the physical allocation of resources. This approach provides enhanced dynamicity in the management of services by the user. The orchestration is now distributed over 5 layers, namely user, services and applications, slices and virtual services, network resources, and technologies, to guarantee greater autonomy and better performance.

Our second proposition is functional. The role of the third orchestrator is the virtual deployment and control (VNOS) of the requested services and their VNFs components by integrating the local and end-to-end QoS constraints. Using simulation, we show the interest of this architecture in order to demonstrate the dynamicity of the user request even in the case of a change in service parameters during a session.

Our third proposal concerns autonomy and the introduction of intelligence within the resource orchestrator. In the case of contention, it has rules to improve the ratio of allowable services. Several rules may be applicable, but these rules may contradict each other. We therefore propose a decision support function, capable of choosing the best rule to apply, based on the evaluation of each rule. This function considers many parameters such as cost, profit, or compliance with the SLA. A numerical analysis demonstrates its interest and its feasibility.

These three proposals therefore contribute to improving the autonomy, dynamicity, and intelligence of network management, a booming field. Orchestration is still in its infancy. AI is gradually penetrating networks, and the use of micro-services contributes to dynamicity. Our research, fundamentally focused on improving existing solutions, aims at the holy grail of "zero-touch".

Abstract for the general public:

5G and IoT networks face an explosion in demand, and therefore become more complex and difficult to manage. Automation is limited and human intervention generates errors. On the standardization side, orchestration, SDN controllers, and network virtualization introduce partial dynamicity. On the research side, user centric services require agility and intelligence. However, the orchestration is monolithic. The dynamicity and autonomy relating to "on-demand" are not guaranteed. Thus, after 20 years of experience in the telecom industry, our contributions attempt to meet these new challenges. Our first architectural and organizational proposal introduces a new layer to design and manage virtual services. Our orchestration is distributed over 5 layers to guarantee autonomy and performance. Our second proposal, functional, supported by a simulation, addresses the dynamicity of "on-demand" services. Our last proposal supported by a numerical analysis, is a decision-making function, based on the high-level SLA in order to improve the ratio of allowable services. Faced with these needs of autonomy, dynamicity, and intelligence of the new ecosystem, our research aims at the holy grail of "zero-touch".

Chapter 1. General introduction

1.1. Context and Motivations

From 1991 to 2011, I worked in the Telecom industry in major and innovative vendor companies in France and Israel. Evolving in the domain of the network management for different telecommunication technologies and in different positions, I have witnessed many changes during these years. In my opinion, the most significant evolutions occurred from 2005: the network management applications joined the data center and the IT world. Their requirements in the Telecom tenders begin to include several new features related to the IT and data center technologies. Thus, a lot of efforts were invested to adapt the network management applications to the new data protection, platform, and security IT requirements. Then, more and more new requirements for the network management came from the IT world and later from the cloudification process.

Moreover, these last years, demand on wireless network for services mobility, for ever more resources, and for a growing variety of services and applications, exploded. The offer of on-demand services has also accelerated. Consequently, network management as we have known it up to now, with its silos and rigid approach, cannot meet these expectations and the growing complexity. If we add the Internet of Things, the network needs to become autonomous (cannot depend anymore on human intervention), dynamic, more adaptable, and flexible. In this context, the entire industry and forums are involved and have specified, and standardized new architectures based on software-defined network (SDN), network function virtualization (NFV), and orchestration. Other new technologies such as micro-services or network slicing have been developed.

How will these evolutions and new paradigms impact the legacy network management systems and the operations support system (OSS) and consequently change the vision of the network management? Which architectures will be able to meet the requirements of autonomy and dynamicity?

These numerous changes aroused my curiosity and explained the choice of this challenging field for my thesis.

1.2. Challenges and Objectives

When beginning the research, we were amazed by the number of companies, standards, and forums involved in the SDN and NFV domains. The research in this field although less

prolific, evolves just as rapidly. After a few months, in order to better understand the different actors, their respective place and influence, and where we will be able to best add our contribution, we understood, the necessity of a survey. This was our first challenge and objective to position ourselves in this very large and fast evolving domain, a kind of entry key. Considering SDN and NFV networks were already well covered and even partially implemented, we decide to focus on orchestration and its influence on OSS functionality and the changes needed to achieve autonomous (also called zero-touch) networks.

Therefore, we better understand the functions and the organization of the future network management system (NMS) and its actual limits. Based on this study, our challenge was to understand how we can transform existing networks into truly autonomous and dynamic networks. Are the current architectures and technologies sufficient to address these needs and this growing complexity?

This reflection led us to think about enhanced elements of architecture and new technology usage to improve the network autonomy and dynamicity. Additionally, more intelligence is necessary at the level of the network management in order to process the large amount of decisional data sent by the network and maintain autonomy in all scenarios. The objective is to solve part of the limitations that appear in the survey. The aim of this architecture and extra intelligence is to warranty the autonomy, dynamicity, and flexibility of the future network management.

1.3. Thesis Scope Problem Outline

At the beginning of this thesis in 2017, some implementations of SDN controller and NFV already existed in the industry. Even if at an earlier step, the SDN and NFV concepts were already defined and specified in the different standard development organizations (SDOs). The orchestration was newer and still hardly defined and once working on its role and functions, the place of the OSS in the future network management must be readjusted. By studying what is done in SDOs and research, we better understood the current concerns: the support for multi-technologies, large network size; the growing demand for service diversity and dynamicity; the accelerated demand for new services and applications; the role and function of orchestration in the proposed SDOs and research architectures and how they influence the current NMS and OSS with its rigid and silo approach, in the future management of the network.

On the new technologies' side, SDN, virtualization, cloudification, orchestration, network slicing, or artificial intelligence (AI) are introduced. In the research, new applications and European projects show the interest of the technologies mentioned above. The service composition with its different elements like links, network nodes, or VNFs, require a digital ecosystem. The introduction of these new technologies tries to solve the current problem of the deployed networks, especially regarding the revolution in the user demand.

This revolution in user demand is reflected on the needs side, by request for personalized, on-demand, and dynamic service and on the network side by greater complexity to manage. All these

last years' requirements together with the complex and growing networks, require dynamicity, integration, scalability, and effective management.

At the intersection of the current concerns, the new technologies, and the need (see *Figure* 1), the following questions arise: Which architecture could guarantee the autonomy (without operator intervention) and the dynamicity of the network? How can we respond effectively to the growing demands and complexity of network? What types of functions and information will give us a smarter orchestration? How can the new technologies help solve these problems?

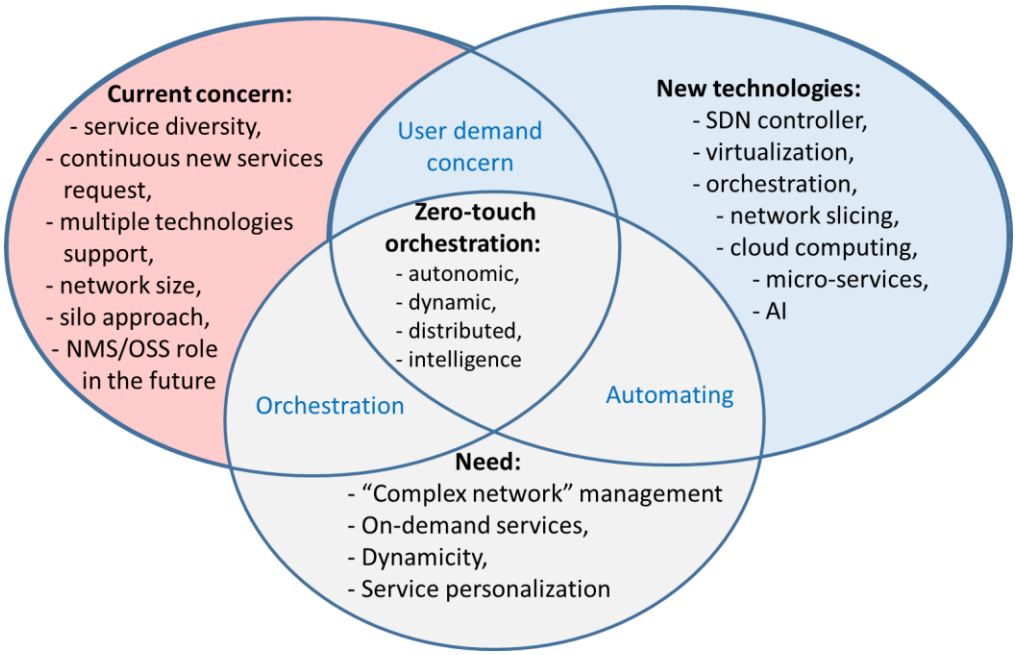


Figure 1: Problem scope

These questions led us to network management architecture considerations for fifth generation of mobile (5G) and IoT networks. We consider the usage of the new developed technologies and the integration of intelligence in orchestration to ensure this autonomy and dynamicity requirements. The aim remains the zero-touch network.

1.4. Contributions

Based on a comprehensive state of the art of the future network management focused on orchestration and its influence on NMS and OSS features, this thesis proposes the three following contributions:

1. An architectural and organizational proposal, related to user centric dynamicity, network dynamicity, and orchestration distribution features, focuses on the two following points:

- An additional network services and slices virtualization layer above the SDN controller to allow virtual service creation with every component without being limited by the physical

resources. This should make possible on-demand component dynamicity without a loss of services when provisioning a service and thus ensuring service continuity.

- A distributed orchestration for the 5 layers: User, Application and service, Network service and slice, Network resource, and Technology. This distribution delineates and isolates the responsibility of each orchestrator and thus allows an improvement of the performance and autonomy of each layer.

2. A functional contribution presents complete use cases with the help of various diagrams of sequences, where the added value of our architecture is illustrated. In this approach, the flow is detailed together with the way the information is updated, and accessible with the help of the interaction between the different functions. This functional approach can even bring to new functions such as more proactive management of the services by the usage of rules and events qualifications. Among different use cases, we choose to simulate one scenario that illustrates the user-centric service dynamicity and network autonomy features, with a real platform and open sources. This simulation demonstrates the service continuity ability when dynamic changes occur during a running service involving VNF modifications.

3. From the previous proposals, the orchestration appears as the decision center where multiple rules are defined. In this context, several rules can be applicable, or some rules may contradict each other. Thus, we define an intelligent and autonomous decision-making function within the resource orchestrator to decide the best rule to apply in case of contention (where the network capacity is not able to support the number of service requests). This should enhance the number of allowable services. The originality of this function lies in the introduction of intelligence at the orchestrator and of rules related to the business, regulation, or policy at the network resources level. This mathematical function also pushes the limits in the use of resource orchestrator rules by its ability to select the applicable rules and choose the best rule considering several parameters and variables. Therefore, this results in an improvement of the network autonomy. A numerical analysis shows how the usage of this decision process creates a more resilient and autonomous network.

1.5. Manuscript Structure

The manuscript includes seven chapters and one appendix and is organized as follow: Following the introduction, the second chapter is based on the state of the art of the main architectures proposed by the SDOs together with a brief overview of the research in this area. With the help of a survey detailed in the appendix, we show how the SDOs, the research, and the new technologies try to meet the needs for an autonomous and dynamic network management. We identify the challenges and gaps in the architecture and functions of the network management.

In chapter three, following the analysis of the limitations of the architectures detailed above, we present a new architecture which should solve some of these limitations. We present the main functions of this architecture and their organization in different modules.

In chapter four, some scenarios and sequences diagrams illustrate the functional proposal and the flow of operations in this architecture. These use cases show the added value and various aspects of our proposed architecture.

In the fifth chapter, we demonstrate the interest of this new architecture with the help of a simulation. We compare it to other architectures that do not include our added elements and try to evaluate how this architecture can demonstrate the dynamicity of the user request even in the case of a change in service parameters during a session.

In chapter six, we focus on the autonomy of this network management and define a decision-making function located in the network resource orchestrator that helps the network to allocate resources autonomously even in conflicting scenarios.

In the last chapter, we conclude our research and its contribution and enlarge its scope to define some future research directions. As written before, the appendix presents a detailed state of the art of the main architectures' approaches in the SDN standardization forums and institutes and discuss the place of the orchestration in these different architectures.

Chapter 2. Faced with the needs, the limits of the existing

2.1. Introduction: description of needs

Studies from IHS and Gartner market analysis companies [1, 2] describe a global mobile network with six billion connected smartphones and around twenty billion connected objects by 2020. According to Statista [3], the number of internet of things (IoT) connected devices worldwide will be 38.6 billion by 2025. In this context, demands for network mobility, scalability, dynamicity, adaptability, and flexibility, grow very fast from year to year and a too-rigid legacy network management as realized with NMS/OSS, cannot meet this demand. Therefore, there is a need for more dynamic network management to answer the need for permanent changes (mobility or service components modification) but also to answer the continuous introduction of new services. Moreover, given the last two decades, the complexity and the size of the current networks no longer allow for human management. Also, customers wish to pay less and less for telecom network infrastructure, perceived today more as a gate where one fee per month should provide access to the unlimited resources of this network. As a result, the operators and the vendors are under great pressure to reduce capital and operation expenditure. In order to face these technical and economic challenges, most vendors and operators realize the need for automation of the network and try to specify and standardize a zero-touch network with complete automation of the different features. In this case, there is no more need for human intervention in the daily operation of the network at the level of the service provisioning and real-time maintenance, but also to assist customers to order their services. This should generate significant savings in cost of labor and improve the performances.

Therefore, SDOs and operators felt the need to change. The NMS and OSS cannot continue to manage the network in a monolithic way and a new approach to network management became necessary. The IT world has faced a similar problem and has operated a separation of user needs on one side (with the cloud), and the infrastructure on the other side. The cloud and micro-services allow the user to be autonomous and to obtain "on-demand" services through an event-driven approach. This event-based approach makes it possible to better distribute the different tasks and each component can perform its function independently. Also, when requesting services such as SaaS or PaaS, this approach makes it possible to better manage resources and allocate them as accurately as possible with the help of orchestrators associated with the desired functions.

Understanding this need for changes, SDOs have introduced SDN, NFV, or orchestration technologies to get out of the sequential and monolithic approach. In the appendix, the survey we carried out on SDOs and research, shows that network management must meet these requirements

with a more event-driven, more dynamic, more autonomous approach, and a new architecture. To find a more suitable architecture, telecoms are looking towards IT with its cloudification and softwarization process that have been pushed by requirements for "on-demand" services. Thus, telecoms are trying to draw inspiration from IT while trying to keep their independence. Both worlds are moving in the same direction. Like the IT world, the network is becoming cloudified and its aim is to offer its network infrastructures as a service: Networks as a Service (NaaS).

With the introduction of SDN and the separation of the Data Plane (DP) and the Control Plane (CP), the network has also begun an automation process. The physical resources are controlled at the CP level to better adapt to the continuous changes in the network and their customers' requests. As generally agreed by most actors in this industry, SDN architecture is split into three planes: the DP, the CP, and the Application Plane (AP). some SDOs like IETF (see Appendix 9.1.2) add a Management Plane (MP) at the same layer as the CP. The automation and softwarization of the network continued with the virtualization of network functions such as firewall, NAT, or wireless node B. This process adds some flexibility to the network and facilitates innovation. However, driven by service providers needing on-demand services, more modular, scalable, agile, dynamic, and manageable solutions, an orchestrator becomes necessary: the NMS/OSS with its silo approach is not able to compose a service within the global ecosystem which can be composed of different virtual network functions (VNFs) and applications with different SLA constraints. The NMS/OSS does not beneficiate of the required flexibility and dynamicity to provision network services (NS) in case of conflict or unexpected scenarios, to apply VNFs in a constantly evolving environment, and to keep track of the overall performance of the network. These requirements justify the introduction of an orchestrator that must be present at the different layers of the network management (physical, network, service, application, and user). Therefore, these new features have to be supported by the orchestration and thus, a large part of the fulfillment and assurance features of the current NMS and OSS will disappear in favor of orchestration (see Table 11). So, the following questions come up:

- In an architecture where SDN controllers, VNFs, and OSSs are deployed, what will be the exact role of this orchestrator within the SDN architecture, its functions, and its interactions with the other entities?
- How will this orchestrator maintain coherent, synchronized, and dynamic network management?
- What are the main challenges of this orchestrator and more generally of the network management of the future?
- Given the new high-level requirements (users centric), what should be done at the architecture level?
- In this global ecosystem, how will the different functions be distributed at the different layers of the network management?

To be able to answer these questions, we must clearly, precise the terms to specify the missing and necessary functionalities. That is why, in this chapter, we describe the main technologies implemented to achieve network autonomy and dynamicity. To illustrate their contribution and

their progress, we use the state of the art detailed in the appendix where SDOs architectures are also explained. We first present cloud computing (section 2.2) and microservices (section 2.3) that come from the computer world and that greatly influence and contribute to the objective of zero-touch network. In the following sections, we detail new technologies that are more telecom specific. These technologies include Orchestrator (section 2.4), SDN (section 2.5), NFV (section 2.6) and distributed orchestration with AI (section 2.7) which are an integral part of our architecture proposal. In section 2.8, we show how the cloud with its "everything as a service" (XaaS) approach is extended to the network management with the NaaS and the network slicing. Finally, we summarize the analysis conducted in this chapter to identify gaps and challenges, and those where we believe we can contribute.

2.2. Cloud computing

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access (user driven) to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. According to National institute of standards and technology (NIST), cloud computing is based on five essential characteristics:

1. On demand self-service: without operator intervention,
2. Resource pooling: virtual resources are dynamically assigned and reassigned according to consumer demand,
3. Rapid elasticity: the user perceives unlimited capabilities,
4. Measured service: control and monitoring are realized by cloud provider but also with minimum operator intervention,
5. Ubiquitous network access: the user can use any client platform, from anywhere, and at any time.

The above characteristics appear to be also crucial for the network management and its new requirements that are also user driven (on-demand services). Three service models are applicable with SaaS, PaaS, and IaaS, standing respectively for Software, Platform, and Infrastructure as a Service. These last years, the concept of NaaS, Network as a service (equivalent of the cloud services for the network), has been introduced. The cloud can be deployed as private, public, hybrid, or community cloud (see *Figure 2*) according to users' requirements.

In the next years, the cloud is extended to include the network devices, but these components are not seen for now "as a service" but are more a way to connect the required applications and functions together (see section 3.3.2).

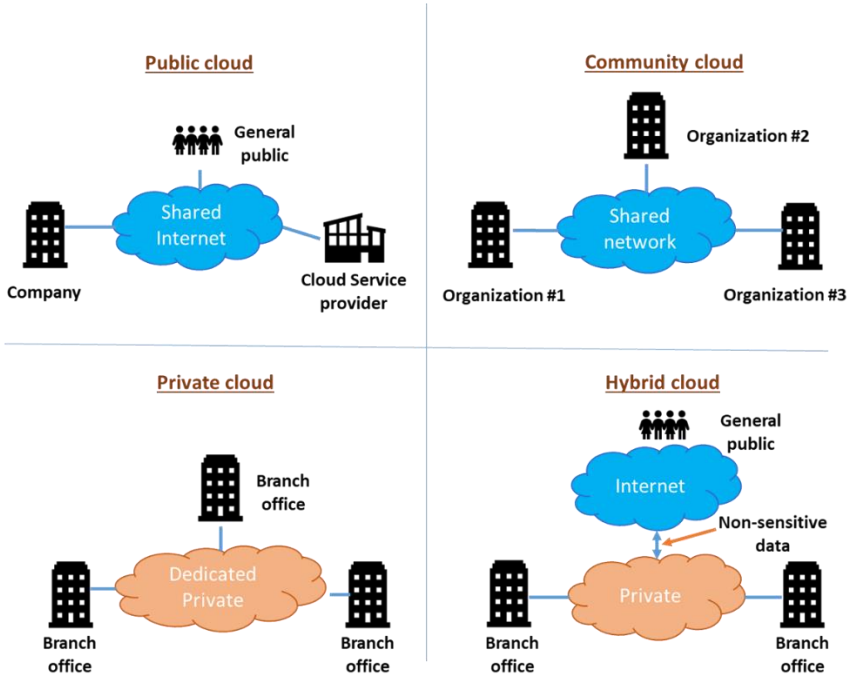


Figure 2: *Different types of cloud deployment*

2.3. Micro-services

Microservices are an architectural and organizational approach to software development where software is composed of small independent services that communicate over well-defined application programming interfaces (APIs). These services are owned by small, self-contained teams. This approach is massively used in cloud computing to provide some flexibility and scalability to on-demand cloud services.

With monolithic architectures, all processes are tightly coupled and run as a single service. This means that if one process of the application experiences a peak demand, the entire architecture must be scaled. Adding or improving a monolithic application's features becomes more complex as the code number of lines grows. This complexity limits extensions and new ideas. Monolithic architectures add risk for application availability because many dependent and tightly coupled processes increase the impact of a single process failure.

With a microservices architecture, an application is built as independent components that run each application process as a service. Services are built for business capabilities and each service performs a single function. Because they are independently run, each service can be updated, deployed, and scaled out to meet demand for specific functions of an application. They can even be implemented with different languages according to the specific needs of the micro-services.

The main characteristics of microservices are autonomy and specialization. Each component service in a microservices architecture is **autonomous** and can be developed, deployed, operated,

and scaled without affecting the functioning of other services. Services do not need to share any of their code or implementation with other services. Any communication between individual components happens with the help of events via well-defined APIs. Each service solves a **specific** problem. If developers add code to a service over time and this service becomes more complex, it can be broken into smaller services. Therefore, microservices benefit of independent working teams, flexible scaling, reliability, reusable code, and continuous deployment (adapted to Agile methodology).

A container is a process created from an executable file, running on a Linux machine, to which certain restrictions are applied. For instance, a container is not allowed to "see" all of the filesystem, it can only access a designated part of it. A container cannot use all the CPU or RAM and is restricted in how it can use the network. Any Linux executable can be restricted, i.e., can be "containerized".

Tools like Docker or Kubernetes allows developers to take their executable, and its dependencies, plus any other files they want, and package them all together into a single file (pod in Kubernetes). Docker also, among other things, allows to include some additional instructions and configuration for running this packaged executable. And these files known as "container images" are also called containers. Container images are self-sufficient. They will run on any Linux machines. Therefore, containerization makes it easier to copy (deploy) code from a developer's machine to any environment.

But the question here is if these microservices can be adapted to networks. In front of the tremendous number of services requests, how can I know in real-time, what is deployed and where, whose services get the microservices, and what happens in case of contention or if some container fails? Some operators already use deployed micro-services and can alternatively activate or deactivate them. In 5G and IoT networks, the Mobile Edge Computing (MEC) moves computing, storage, and networking resources from remote public clouds closer to the edge of the network. Therefore, it serves low-latency communication. Thus, mobile clients can request virtual resources within the access network and respect the local constraint of low delay. But what is about migrating micro-services according to user demand and service composition? These questions are currently discussed in ETSI MEC and in the IEEE conference [4], Barbarulo realizes a Proof of concept of telecom application relocation with the help of container migration. In this domain, the orchestration is often referenced as a potential efficient micro-service manager (VNF manager) to allow real dynamic composition of service. The implementation of a digital system to support dynamic service composition with different components such nodes, VNF, and links is still at the level of proofs of concept (POC) and specifications in SDOs. However, respecting the global E2E and local constraints for dynamic service respect evolving during session remains a challenge.

2.4. Orchestration

As explained before, the orchestration is considered as a key component of the creation of a dynamic and autonomous network. Network orchestration is a policy-driven approach to network automation that coordinates the hardware and software components to support countless demands of services and applications. An important goal of orchestration is to automate the way network requests are handled and minimize the human intervention required to deliver an application or service even in case of conflict or specific scenario. This orchestration is only possible with the help of continuous network notifications, information, and statistics. Given the complexity of today's networks, big data is gradually appearing. Thus, this data feedback needs to be qualified, to allow relevant decisions at the level of the orchestrator. Therefore, orchestration platforms need to be network-aware and use analytics to decide where specific resources should be deployed in order to maintain optimal network performance.

To understand the difference between orchestration and network automation, automation usually refers to the automation of operator / service provider networks, data centers, clouds, information technology (IT) systems, processes, services, or service delivery. This can be defined as the elimination of well-defined repeatable manual tasks. In this case, they all can be automated with proper tools such as automatic scripts. The easiest way to understand orchestration, is to look at it as the grouping of automated tasks in coordinated workflows. The orchestration can modify the order of the automated tasks, select different tasks according to defined rules and different scenarios. The orchestrator should be able to express the intention of the operator when planning the network and its behavior. While it is not mandatory to have the tasks in coordinated workflows be automated to achieve orchestration, automated tasks widely simplify orchestration. These requirements for the network management cannot be performed by the OSS/BSS. Thus, the orchestration has to be autonomic, i.e., "automated" to automate well-defined and periodically repeated tasks, and "autonomous" to make decisions to maintain the integrity, the synchronization, the adapted workflow, and the autonomy of the network management without human intervention.

However, for now, in most SDOs (see Appendix 9.5) the orchestration is often perceived just as an advanced controller, which has to care of every layer from the network resources to the services and user requests.

2.5. SDN

Software-Defined Networking (SDN) is a network architecture approach that enables the network to be centrally controlled, or 'programmed,' using software applications. This helps operators manage the entire network consistently and cleverly, regardless of the used network technologies. SDN allows a separation of the data forwarding at the physical network devices level, and the data control that specifies behavior (see *Figure* 3). Standard APIs such as OpenFlow™ are used to transmit network and devices information and states to the SDN controller with the help of the network operating system (OS). Therefore, SDN controller can effectively replace the routing algorithms of the routers by centrally creating the routing tables and

thus, simplifying the physical devices. SDN approach is also applicable to other technologies such as optics or wireless networks.

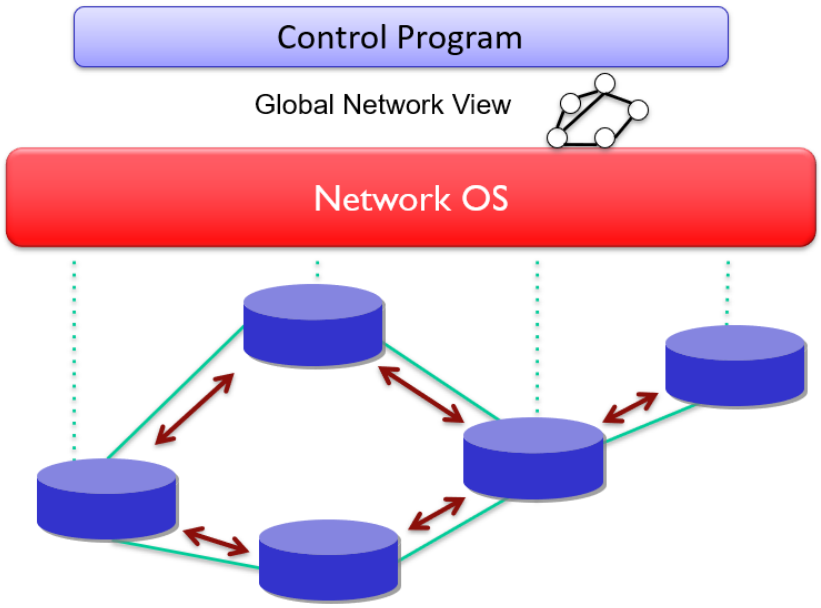


Figure 3: SDN, the different layers of the control plane [5]

SDN benefits of four main advantages:

1. **Network behavior automation** to be controlled by the SDN software that resides above the physical networking devices.
2. SDN built on logically **centralized** network topologies, enables **intelligent control and management** of network resources. Legacy network control methods are distributed where each router sees only its neighbors or the routers of the same Autonomous system. Devices function autonomously with limited awareness of the state of the network. With an SDN-based control, bandwidth management, restoration, security, and policies can be optimized to better adapt to the continuous changes in the network and their customers' requests.
3. Applications and SDN controllers interact with the network through **open APIs**, instead of management interfaces tightly coupled to the hardware. Between the devices and the SDN controllers, OpenFlow, Netconf and Yang are often mentioned, where REST API is used between the applications and the SDN layer.
4. SDN controllers are **easier to maintain,** update, or upgrade because they are SW based running on common servers: commercial off-the-shelf (COTS) hardware like x86 servers. The routing protocol is not running on FPGA integrated circuits that are more complex to modify.

However, SDN controller is mostly limited to routing and path calculation functions. It does not allow flexibility at the level of the functions running on specific components if some modified local constraints of delay, bandwidth, or QoS appear.

2.6. Virtualization – First HW component virtualization

Along with SDN, virtualization is mainly specified and pushed by the ETSI forum (see Appendix 9.1.5). Virtualization, applied to telecom, is the process of creating a software version of network hardware appliances such as firewall, sniffer, DHCP servers, DNS servers, or other end systems. So, there is no longer any need for dedicated HW depending on specific suppliers. Instead of installing expensive proprietary hardware, service providers can purchase simpler switches, storage, and servers to run VMs that perform network functions. This allows to integrate multiple functions called virtual network functions (VNFs), into a single physical server. The other advantages of the virtualization are as follows:

- VNFs are deployed when required and therefore allow a more efficient use of the physical resources of the VMs (CPU, RAM, and storage).
- This generates cost and energy saving (no more need for application on specific HW where 10% of the capacity is used) and deployed VNFs according to the demand.
- This makes the management easier, with user functions mobility, portability, and thus simpler backup, recovery, and replication when needed.
- New applications, SW based, are easier to introduced.

NFV refers to the virtualization of network components, while SDN refers to a network architecture that injects automation and programmability into the network by decoupling network control and forwarding functions. In other words, NFV virtualizes network infrastructure while SDN centralizes network control. Combined, SDN and NFV create a network that is built, operated, and managed by software.

However, because the VNFs are running on COTS, the performances may be worse than with a of dedicated appliance. The virtualization is mostly considered at the component level and not at the network level. For now, only activation or de-activation is possible in this context and VNFs migration based on user demand is still in its infancy. The SDN and NFV coordination is not easy to achieve. This is the reason why orchestration is required. If some dynamic change occurs during a session and a VNF has to migrate to another VM, how can we allow this change during service especially if this VNF is not deployed at the required location? And in such a case, how can we warranty the SLA of the E2E service with its local constraints.

2.7. Distributed orchestration, and AI to assist orchestration

As described in section 2.4, the orchestration is not only an automate but a coordinator of the HW and SW components of the network to ensure the E2E service provisioning considering the local constraints. So, the orchestration has to operate at different levels: the user, the network, and the physical resources at least. Each orchestration should then be able to operate independently to warranty the performances of each layer. However, except the TMF that has specified four levels of orchestration with the technology, the resource, the service, and the user layers, every other SDOs have a monolithic approach with one orchestrator at the network resource and service levels (see Appendix 9.3). Indeed, it should be possible for an orchestrator to

operate at the technology level, for instance to optimize the device usage and reduce energy consumption. A resource orchestrator should be able to do the same at the resource level to identify some missing VNFs in some specific locations or missing resources on a VM. At the network level, a network/service orchestrator should be able to respect the E2E service SLA together with the local constraints (see Appendix 9.2.3). The ability to overcome unforeseen scenarios like resource conflicts, security threats like DOS attacks, fault detection, or fraud (see section 6.6.3) can also exist at the different layers of the network management. This is similar to the role of conductor in a real orchestra, which is certainly the most sensitive function. Because of this, the orchestrator can become a bottleneck for every new service or VNF. The challenge stands also at the inter-domain and multi-layer technologies [6] and requires coordination of different vendors. The question of scalability [7, 8] is thus crucial because all these parameters may generate excessive control traffic overhead. The performance or the notification entities will have to process a tremendous amount of information to serve the different applications such as the OSS, BSS, the orchestrators, and the controllers. Efficient monitoring applications with responsive interfaces are necessary to achieve scalability of these networks. In the same vein, the application orchestrator has to be multi-domain in order to manage every kind of services and VNFs coming from different SP and customers. This complexity and this quantity of data translates into the need for a certain intelligence, located at the orchestrator level, to effectively process this decision-making information. AI is now often cited, and even if still at its infancy, the ETSI forum with GS ENI [9] specifications of 2021, clearly defines how AI can enhance the performances of the orchestration and the network management.

Considering the number of layers and modules, the on-demand and E2E services require the specifications of clear APIs to allow the ISPs to cooperate and provide an E2E service even if it has to go through several operators. In this case, we speak of horizontal interfaces as defined in MEF with LSO (see Cantata, or Sonata APIs in Appendix 9.1.3). But as described before with OpenFlow (see ONF in Appendix 9.1.1), NetConf, or REST, vertical APIs allow standard communications between the data plane, the control plane, and the application plane.

The introduction of SDN, virtualization, and orchestration allows to be less network-centric and to position the user at the center. Together, these technologies must allow service customization (personalization) in theory and support for dynamic services (with possible modification during session). However, despite long discussed in the forums but even if good concept, and well specified, the orchestration remains a monolithic element often defined as an advanced controller.

2.8. Cloudification of the network with Network slices and NaaS

As the services proposed in the cloud with the XaaS models, telecoms want to draw inspiration from the computer world and extend these models to the network where the network will be proposed as a service (NaaS). This concept brought to network slicing.

Network slicing as described in Duan's book [10], is born out of the need to make several networks coexist with different requirements on the same physical network infrastructure. For example, an

urban tele-surveillance network and an augmented reality application do not share the same needs as a network dedicated to remote health, or a network of 5G mobile communications. In reference [11], for critical machine communication (CMC) network for transportation such as autonomous cars, working groups propose an additional and separate secured network slice with its own capacity, extremely low latency, and availability requirements within the future network. This network slicing supposes a network architecture that enables the multiplexing of virtualized and independent logical networks on the same physical network infrastructure, each one with their own monitoring and controller OS. These virtual network operating systems (VNOS) per slice will make them easier to manage. Each slice is isolated and secured and sets up an E2E network following the diverse new services and application requirements. For this reason, this technology plays an important role for the future networks that are designed to efficiently support a huge number of applications and services with widely varying SLAs (see Figure 4). The concept consists in realizing the implementation of flexible, dynamic, and scalable slices on top of shared physical network architecture using SDN, NFV, and orchestration to create new business opportunities [12].

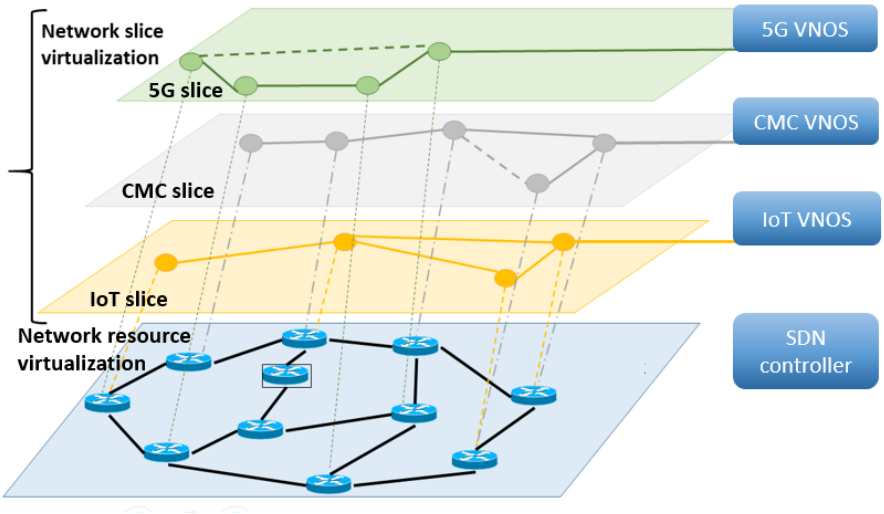


Figure 4: Different slices coexistence at the virtualization map layer

On top of these network slices, specific services can be provisioned. Because they exist on the slice before being deployed, this may provide an additional component dynamicity to the on-demand service by virtually deploying services before allocating the resources in the physical infrastructure. This may allow a step forward to dynamicity of user-centric and on-demand services. However, for now, this technology remains mostly at the level of standardization, research, and proof of concept and there is no specific controller per network slice and service to take care of the E2E and local service constraints.

2.9. Conclusion: Gaps and challenges

Today, the objective of complete automation, zero-touch, self-healing, self-optimization regarding availability and resources consumption, in order to be able to meet the 5G and IoT

challenges, seems yet far in the horizon. The orchestrator appears to be a central piece of the network management of the future, but still needs clear specifications and convergent definitions. Many POCs have been realized, together with very important operators' initiatives like open network automation platform (ONAP) and Central Office Re-architected as a Datacenter (CORD) (see Appendix 9.1). But in these projects, the orchestrator appears more as a stronger SDN controller for multi-domains or application of rules. So, what are the main issues and challenges we can identify?

As explained before, the horizontal and vertical interfaces need to be well specified to allow real interoperability of the different application orchestrators. They should be capable of providing relevant information to OSS, BSS, and management applications, if there is a need for human intervention or for component extension. The limited role of OSS and the NMS in the future is also explained in the Appendix 9.4. But the implementation of these functions is more suitable for SDOs and operators who have to find the right migration path to the future NMS/OSS and specify together standard, open, and adapted APIs for the needs of the industry. This is a long-term task. As detailed previously, new technologies play a strategic role to support autonomous and dynamic requests for telecom networks. But some gap and challenges still require attention as described in the *Table* 1:

Feature	SDOs, research, and new technology	Our concern
Autonomy	Consciousness of all the actors of the research and industry. Automation with API for user-centric service with Cantata of MEF LSO without need for operator intervention. In TMF, there is a requirement for service continuity & optimization, or self-healing and ETSI has specified an AI with GS ENI document.	√
On-demand service	MEF working on supporting E2E on-demand service with the specifications of Cantata API at the business and NMS level for multi-domain support. some POCs were already realized.	√
Network dynamicity	Discussed at the MEF level. But still limited in the facts. ETSI want to use the GS ENI AI to assure part of this dynamicity. In the research, some articles speak about constraints dynamicity to be considered for instance in case of user mobility.	√
Distributed orchestration	Clearly required at the TMF level. In most SDOs and implementations, the approach remains monolithic and mostly centered at the resource or network level.	√
Intelligence	Intelligence required to qualify decision-making information, maintain autonomy in case of conflict, contention, or unexpected scenarios. This can be used also in case of security attack, fraud, or fault detection.	√

Table 1: Gaps and challenges

For Autonomy everyone is aware of this need. Several implementations and new technologies provide a partial answer like the control loop (see section 3.2.2), the use of rules in the orchestration, or the cloudification of the network.

On-demand or user-centric service requires dynamicity at the level of API to assure interoperability between the different SP who can be involved in the required service, but also at the level of the architecture to warranty dynamicity of the service components even during a session. Work in the various SDOs and research remains network centric and not user centric enough. What happen if a user requires service dynamicity, i.e., wants to modify its service during a session? This will mean for most ISPs to stop the running service and provision a new service with the required modifications (if supported by the SP). This dynamicity has also to be supported at the network level.

The role of coordination between the different applications and layers is realized at the orchestrator. Therefore, as explained in section 2.7,one centralized orchestrator may limit the network management and may not be able to best meet each layer needs for orchestration, performance, and autonomy. Thus, distributed orchestration may assure these functions.

Decision-process is always cited in the different forums as a rule-driven decision making and/or as an artificial-intelligence-based algorithm. However, few papers cover this subject. How will this orchestrator be able to warranty services request for specific delay, jitter, and broadband with an acceptable response time? Sometimes, by reading different documents and articles, it seems all unsolved problems at the SDN layer are relayed to the service orchestrator… In case of conflicts between applications and services contention, how many rules will be necessary to give the right priority to the right application? Decision-making algorithms and real-time consideration are often contradictory: there should be a trade-off between algorithm complexity and real-time requirements. Therefore, intelligence is required. Deep learning [13], has already shown very impressive results with autonomous cars or medical diagnosis in the last years with its advanced algorithms. However, it is resource consuming, and no one really knows how these advanced algorithms "do what they do". So how can we keep control and diagnose some defect by using such algorithms to get to zero-touch networks?

Security is also a strategic issue. As the brain of the network is now software based and should be located in the cloud [14, 15], every transaction between modules, will require authorization, authentication, and even encryption in case of inter-domain transactions. Every module will require built-in security. But this issue is not covered in this thesis.

To summarize, we have chosen to focus on the problems of user and network-centric, distributed orchestration, and intelligence introduction to warranty better performance and autonomy of the network. This should provide efficient orchestration of the network management and dynamicity to provide SP more autonomous and user centric architectures. These features appear as strategic for the entire industry to achieve programmable and autonomous/zero-touch networks and will be the basis of our different proposals. In the next chapter, we propose to address these important issues with an enhanced architecture for future network management.

Chapter 3. Architecture and organizational proposal

In Chapter 2, we identify the gaps and challenges that exists between the required features and the current implementations of the network management of the future. In this chapter, we propose an architecture that will solve some of these gaps.

3.1. Introduction: questions about current architectures

5G and IoT telecommunication networks are required to manage critical, considerably diverse, and constantly changing demands. For critical machine communication network for transportation such as autonomous cars, the requirements will be very different of hologram or augmented reality network slices. In Chapter 2, after description of the needs and new technologies based on a survey detailed in appendix, we have understood that dynamicity and autonomy are required for the network management of the future to achieve the zero-touch network. Despite several SDOs and research have proposed different architectures to support these requirements, the following gaps and challenges were identified: the network will have to be autonomic, to support user-centric services; this means that the network management will have to support dynamicity to respond to this on-demand service requests; The complexity of the networks and the tremendous amount of data that will be generated, justify the use of a distributed orchestration and the introduction of some intelligence to make the right decisions in all scenarios and maintain the autonomy of the network. These considerations lead us to the following questions:

- What is missing to these current models and architectures to fit these gaps? What has to be added or modified for our proposed architecture to fill the need for autonomy and dynamicity?
- How to guarantee the need for on-demand services when these services evolve even during a session? How should the different functions interact to allow a continuous service flow without disruption?
- How should we define and organize orchestration to answer the need for independent and autonomic behaviors at the various layers of the network?
- Where and what kind of intelligence do we have to introduce to process the massive amount of network decision-making information and handle conflict situations and dynamic on-demand services?

We believe that dynamicity must allow and support different and evolving assemblies to be provisioned during a session, in order to meet modification requests without requiring total redesign of the network service. By autonomic requirement support, we intend to reach a zero-touch network, by using an effective organization of orchestrators. In this chapter, we will present some answers by proposing an architecture and organization proposal for modern network

management to get closer to the objective of zero-touch. In Section 3.2, after recalling the main points of the current architectures, we state the resulting problem. In Section 3.3, we detail our proposal: a novel architecture that allows dynamic changes during a session, and an organizational model that describes the distribution of the orchestration to allow improved flexibility, scalability, and performance, to ensure network autonomy. In the conclusion, we summarize the advantages of this architecture and organizational model.

3.2. Problem

3.2.1. Main points of current architectures

Network Management SDOs architectures, such as those mentioned in appendix commonly include the following points (see Figure 5):

- A clear separation exists between the control plane and data plane, providing more adaptability to the network. For instance, in a network of switches, if the bandwidth requirements change, the SDN controller together with the orchestrator is able to adapt the running services in order to satisfy these modifications without involving the network management function. This is what we refer to as adaptability in Figure 5 between the SDN and the network equipment virtualization.
- Network Equipment virtualization, VNFs, and VM have been introduced, even if not always required for every telecommunications technology.
- The orchestrator appears as a decision and coordination center, generally located at the service deployment layer between SDN and the OSS/BSS. It can modify and program the SDN controller behavior to warranty the autonomy of the network. The orchestrator also interacts with the VNFs and can request to move a VNF to another VM for instance.

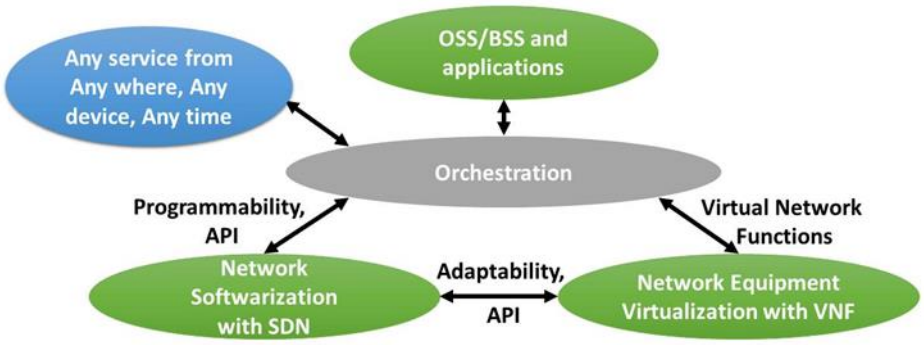


Figure 5: Main modules in Orchestration and Management architectures

The aim of these network management architectures is to satisfy the need for the "4Any" requests: Any service, from Anywhere, from Any device, and Anytime [16]. Once supported, is this 4Any

service still available in case of any dynamic changes within the same session? Are these current architectures enough to reach a zero-touch or autonomic.

3.2.2. Problem: architectures weaknesses

Our proposed architecture is motivated by the following problem. According to the scientific literature and main SDOs, the way SR constraints are considered at both the application and resource levels, is partially covered, especially in case of dynamic changes (see Appendix 9.2.3). When, during a session, the applicative components should be added or migrated dynamically (on demand), the actual implementations cannot assure service continuity, and autonomy can be threatened. The service can be lost and must be redesigned, provoking delay and QoS problems. We can cite here two different cases that the existing architectures cannot manage:

In the first case, an end-user watches a film. This user is connected to a video streaming server. Then, the link to the server becomes overloaded. Thus, the network has to find another video content server with the same film and connect the user to an alternative second server so that the user can continue watching the film. In most cases today, the link is simply lost when the video quality becomes too bad and the user must find another server with the same film, manually, with the help of a search engine. If the video content server is protected and sold as a high-quality service by a SP such as Netflix or YouTube, some specific SP application are implemented to switch the existing service and connect to another server with the same video and continue from the point the film was interrupted. The only difference is that the video content may be physically located further from the customer. This usually results in service degradation for the user, with some delay, and maybe a service interruption before recovering. Moreover, some specific solution at the SP application level has to be implemented including load-balancing to answer this problem.

In another case, during a VoIP conversation, an end-user may want the operator to physically E2E encrypt their call. For instance, they want to transmit their credit card number for payment. This is an on-demand request for dynamic modification of a call that cannot be realized in the same session at the network level by current architectures. This may be demanded only at the application layer if the two persons who communicate have the same application that supports this feature. It is worth mentioning that with an architecture such as ONAP [17] that is based on ETSI MANO, the closed control loop and other real-time monitoring modules allow calculating another path and modifying the running service. However, the addition of another VNF function such as encryption, with its queuing and processing delays, may generate serious service degradation. Using a control loop is not a warranty that the E2E QoS constraints will be respected. At least several iterations of the control loop will be necessary.

In the next section, we detail our architecture proposal and show how it helps resolving the limitations of dynamicity and autonomy. Rather than dealing with specific problems like self-healing, optimization, energy saving, or decision process, we opt for a holistic approach including architecture, and organizational proposal.

3.3. Architecture and organizational proposal

The objective is to provide enhanced architecture to solve the E2E and local constraints while creating or dynamically modifying a user-centric service and maintain network autonomy. In this context, we consider the following questions: at which levels of the architecture can these problems be identified? Which modules must be added or modified? What are their interactions? Can this avoid resetting the complete service?

3.3.1. Architecture proposal

Our architecture tries to synthesize some research concepts regarding NFV and network slicing in the context of application and resource constraints along with the work realized in SDOs. In our proposed architecture, we represent at the CP level, an aggregation of several SDN controllers under a multi-domain, multi-technology controller that is orchestrated at the technologies and resources levels (see Figure 6). TMF among several proposed architectures, details in [18] every NM layer and separates the orchestration into four distinct levels: user, applications and services, network resources, and technology orchestrators. They can communicate with each other via API such as REST. With this functional distribution, each orchestrator is responsible for one NM level. Once the work of an orchestrator is complete, it does not have to be done again at a lower one. Further, each orchestrator trusts the work and decisions of the orchestrator above it. For instance, the second orchestrator selects the application server. The following ones will go on building the SR accordingly. As in ISO model, it can contribute to independent layers and interoperability of these layers.

To understand if this distribution in four orchestrators is enough to answer our concerns about dynamicity and autonomy, we analyze the roles and responsibilities of these four TMF orchestrators as follows. Process distribution usually allows improved flexibility, scalability, and performance. In addition, orchestration should allow efficient distributed decision-making process according to the different NM layers. This distribution led us to understand to deal separately with E2E constraints and network resource constraints: one at the network orchestrator and the other at the resource orchestrator. The need for this separation is also mentioned in [19] by pointing the problems in combining SDN and NFV into a unique architecture framework for service provisioning. To this end, this study proposes a two-dimensional abstraction model where SDN is related to split into planes concept, and NFV to the five network layers concept. It proposes an additional dimension of abstraction to decouple service functions and network infrastructures. Therefore, we realize the necessity of a multi-layer NS to allow user service transparency from one side and network resource continuity from the other side: a type of virtual NS composer (planning component). This allows the creation of alternative virtual service if required but, without removing the running service until the modified one is ready. Consequently, after the creation of this new virtual service, the running service can switch to it. This explains the need for an additional virtual layer.

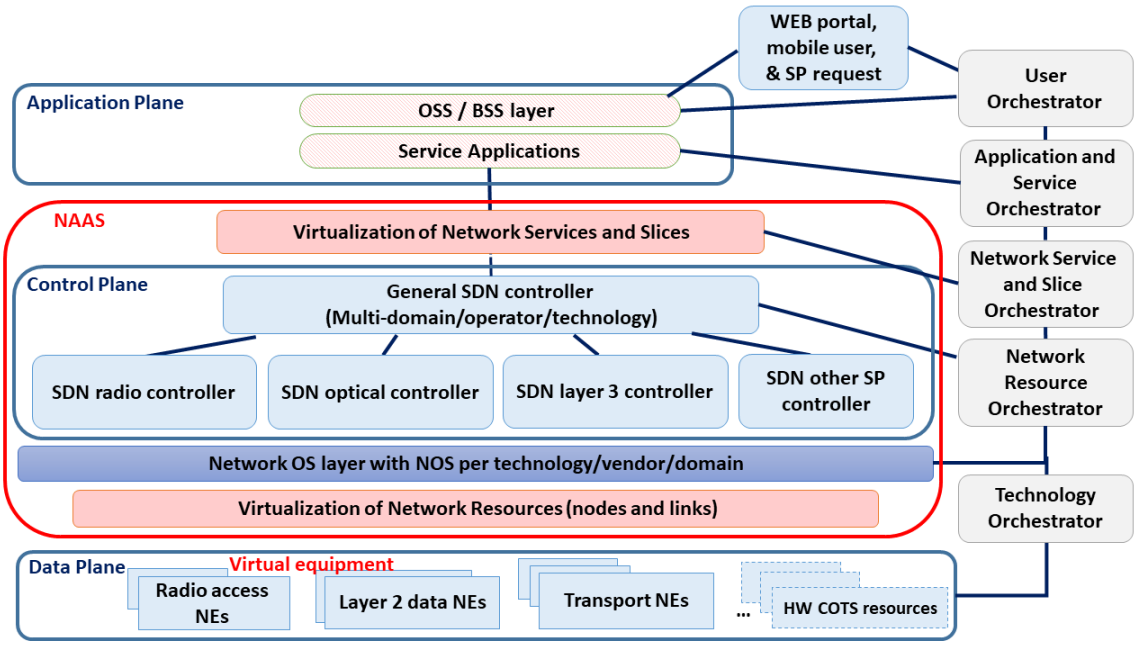


Figure 6: Proposed architecture

Therefore, in the Network-as-a-Service (NaaS) part, we introduce a fifth orchestrator: the "network service and slice (NSS) orchestrator", related to the additional virtual layer and responsible of service and slice attachment. Service attachment means service assignment without resource allocation: this is a virtual service deployment. This orchestrator operates at the network level, and it is responsible for design or re-design of a SR. This SR "design" (global virtual deployment) includes the main components of a service (nodes (VNFs), links, and endpoints) with its application's QoS constraints. In MEF, the multi-domain aspect between cross-partners is covered, at the network level, at the service orchestrator using Interlude interface. Likewise, the NSS orchestrator will ensure this role. It should be noted that this orchestrator is a split-off from the network resource one. The latter works directly with the general SDN controller (multi-domains, multi-operators, multi-technologies SDN controller as described in the Figure 6), which is responsible for SR placement using the network resource virtualization.

This new NSS orchestrator cooperates with an additional virtualization layer: the "network services and slices virtualization," located at the northbound interface of the NaaS. This virtualization layer is an abstraction of the network resources. For instance, a group of switches in a business building can be represented as one entity with its general capacity, delay, and constraints. As explained in [10], tight coupling between service provisioning and network infrastructure quickly becomes a barrier to rapid and flexible service deployment. With the introduction of this network virtualization layer, the SR attachment is created in two phases. First, at the service level it is built with its requirements coming from applications: nodes (applications) links (transactions), endpoints, and E2E constraints. Then, this built SR distributes its component functions over the network abstraction considering nodes (VNFs), links, and local constraints. This selection gives rise to a negotiation. Every service type and VNF are dimensioned and calibrated

with their properties and constraints: throughput, maximum error ratio, and transport time. These service components are registered in the NS catalog that includes E2E service constraints and in the VNFs catalog with more local constraints as in ETSI MANO architecture. For instance, a call service cannot overcome delay of 150 msec. or a virtual NodeB must be in the building closest to the antenna it serves. Therefore, in this attachment phase, the service is virtually built and optimized, by ensuring that the aggregation of the local constraints does not exceed the global E2E service limitation. If a physically allocated VNF does not respect its local constraints, it can be physically migrated to another VM. Consequently, virtual deployment and physical placement are separated and entrusted to well-identified and independent managers, each having its own constraints to respect. This "virtual deployment" respects the global SLA regardless of the physical resources.

Then, the attached SR is sent to the general SDN controller and VNF manager for physical placements. The general SDN controller controls different technology and domain SDN managers that can be viewed as virtual networks with their own controllers [19]. Therefore, this network virtualization layer can impart more dynamicity to the network. The advantage of this network virtualization layer is the ability to pay special attention to the global E2E SLA constraints and to build an alternative equivalent service without removing the existing one until the alternative is ready and can be switched on in case of modification. This virtualization layer, unlike the previous upper layer, does not consider the SP services and server locations; it is centralized on the network itself. For instance, in this layer, each network slice request is independent of the others. In case no alternative slice is found, this can be reported to the OSS layer for planning a network extension, e.g., the slice can be scaled up to a slice with more capacity.

Once the service is attached, the general SDN controller translates and calculates the services requirements. For instance, the SDN controller for switches' technology continuously calculates its routing information bases (RIB) and updates the forwarding information bases (FIB) of each switch of the network, for packet forwarding. In addition, the VNF manager looks for the VNFs associated to the service request, with respect to SLA and its QoS general constraints. For instance, a general application constraint can be to not overcome a determined number of n hopes (switches) to respect the maximum delay.

Then, the different SDN controllers of Figure 6 use the API of the network operating systems (NOS) to control and monitor the different network white-boxes. This is done per vendor, per technology, per equipment type but also per domain (virtual network). At this level, there is a network resource virtualization including links and nodes, which is an abstraction of the equipment of the data plane.

BSS receives statistics from the network for billing and business applications. At the application plane, OSS still appears in main SDOs architectures with a limited functionality such as setup, optimization, or long-term functions [GS1].

By creating a virtual service, we express the requested QoS constraints for the IT resources. We express the local and E2E networking constraints for this SR, i.e., the offered QoS NSs with the

help of the NSS virtualization layer. The attached service is a result of the negotiation between the requested QoS constraints and the offered QoS NSs. While addressing the SDN controller, the SR is already virtually deployed with its E2E constraints. Therefore, we can summarize the contribution of our proposed architecture as follows: we add an NSS virtualization layer for NS virtual deployment thereby, allowing loose coupling between SR attachment and physical placement. We separate the resource orchestrator into two functional entities.

3.3.2. Cloudification of the network

To better understand the origin of our proposal, Figure 7 explains how the service is built in main SDO platforms and cloud computing and how it is done with our proposed architecture.

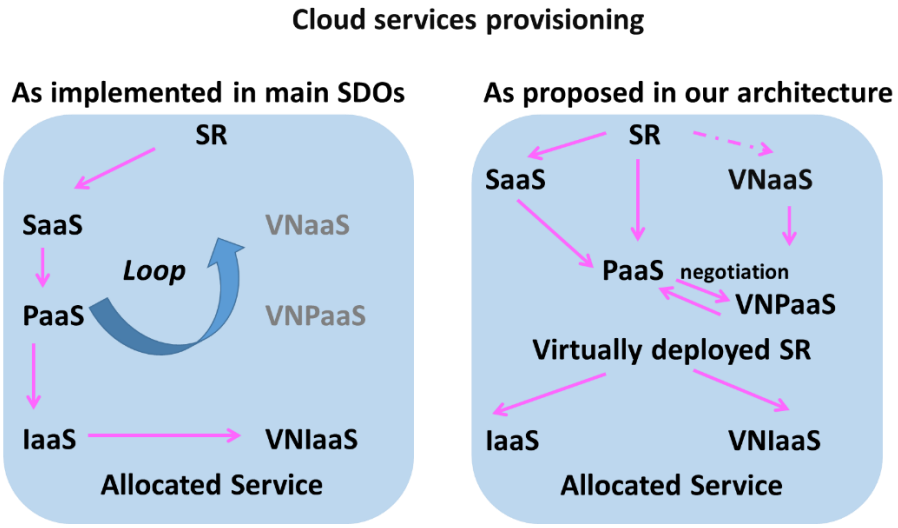


Figure 7: Comparison between service request with SDO approach and with our architecture

In cloud computing, according to functional analysis, three cloud service components are used to provision a SR: Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS) [20]. In reference [21], cloud computing is defined as "a style of computing in which dynamically scalable and often virtualized resources are provided as a service over the Internet". Cloud computing is based on the evolution of technology innovations in virtualization, distributed computing, utility computing, grid computing, Web services, SOA, etc. It aims to provide IT capabilities (computing power, storage, application functions) in the form of on-demand services to the end customer through easy-to-use Web technologies.

Provisioning cloud computing SaaS (see Figure 7, left side), requires the following steps [21]: in case of SR assisted by an account manager of the cloud computing provider, the end-customer must register first in the cloud computing provider platform and specify his need and budget, for instance in a case of e-Commerce as a service. The customer must select the applications and functions he needs for the SaaS. Then, according to the importance of the customer service (number of clients, number of products, of applications…), virtual resources and OS (PaaS) are chosen. The relevant infrastructure (IaaS) is then selected and activated on different servers respecting delay constraints in case of several applications or functions running together. Finally,

the networking part (to connect the customer and its applications) is selected with the help of virtual network IaaS. Although this approach can be optimized, it remains essentially vertical. If strict delay constraints exist, the connection of the different applications and functions of the SaaS, may be problematic with this approach.

In the telecom world similar analysis was carried out to "cloudify" the network with virtual network: VNaaS, VNPaaS and VNIaaS [10]. VNaaS can be used when there is a pure network service request such as a network slice. VNPaaS offers a platform to provide VNFs with programming tools and virtual links. This model equivalent to cloud PaaS, represents virtual network services and proposes a service platform with catalogs of exposed services. It thus allows users to choose and compose VNFs like NAT or encryption service. VNIaaS consists in the virtualized physical network infrastructure and is equivalent to NFVI in ETSI architecture. As cloud IaaS, this model offers resources environments for VNFs like computing and storage but also networking capabilities for functions like routing and switching.

In the unified networking and cloud computing, theoretically, there should be a cooperation, a horizontal approach and coordination between the networking and computing systems for E2E service provisioning. As the services went through cloudification, the network part also goes through the same process. According to equivalent services models, the SR is first addressed to the SaaS and VNaaS to first define the requested provider's applications and the service types with global network parameters. Then, The SR is addressed to the PaaS for applications and VNFs and to the VNPaaS for networking VNFs and virtual path to connect the applications and VNFs that compose this SR with their different constraints. This step should express the need for resources on virtual machines and, their QoS and relative locations constraints. this SR should then be sent to the virtual infrastructure level to find the physical network and computing resources satisfying the E2E and QoS constraints of the service and its components, and to finally allocate the service physically.

However, in the SDO open-source platforms like ONAP or CORD, the IaaS addresses its request directly to the VNIaaS without using the VNPaaS. A control loop mechanism enhances the difference between the calculated service and the required service, especially concerning the simultaneous respect of the E2E and the local constraints, with the help of service permanent testing and monitoring. This remains a vertical approach.

According to our proposed architecture (see Figure 7, right side), the SR can be addressed to the SaaS (service including provider's applications), the PaaS (if only need for software platforms), or the VNaaS according to the type of required service. In case of cloud application service, the SaaS directly addresses its request to the PaaS and the VNPaaS. The VM components that require network resources to be interconnected, interact with the VNPaaS. At this step, there is a negotiation to respect the VNF constraints together with the services E2E constraints to define the attached service (virtual service deployment). Once the service is virtually built, it should be sent to the virtual infrastructure level (IaaS and VNIaaS) to find the physical network and computing resources satisfying the E2E and QoS constraints and to finally allocate the service physically. The virtual (attached) service results from the PaaS and VNPaaS. The allocated service results from the

IaaS and VNIaaS. There is a degree of freedom between these two layers: one operates at the virtual network layer while the other operates at the virtual resource level. Therefore, the service provisioning and allocation is realized in two steps and there is a low coupling and a clear separation between the virtual service deployment and the infrastructure allocation. In case of pure network resources request, the network platform as a service (VNPaaS) can be called by the PaaS and by VNaaS. This approach respects the services model and parallel between the computing and networking systems for an E2E cloud service provisioning. We will see in section 3.3.3 that this approach provides some additional dynamicity and autonomy especially in case of dynamic component modifications during a session.

3.3.3. Organizational proposal

In this section, we describe two scenarios of service setup and dynamic change, to understand how the different modules of our proposal interact and complement each other. By this approach, we can better identify the decision-making process in the different orchestrators and the importance of the NSS virtualization. In this context, we define these two scenarios with their different steps, until the service is allocated or modified.

The 5G/Web end-user or the SP requests new services or makes changes to existing ones, going through a portal. This user request is accepted when it is authenticated, authorized, and respects the SLA (see Figure 8, step 1). The role of the user orchestrator is to qualify a user request in a standard and declarative format that is understandable by the second orchestrator with the help of the policy agent. It can also prevent over-subscription via admission control process. This is done today at the BSS level and only a few SDOs see this orchestrator as necessary.

The Policy agent can include some decision-making rules in this process. As specified in the ONAP architecture, these rules and policy agents are distributed on different modules and orchestrators. In the user orchestrator policy agent, an example of rule can be a time manager rule, where some service types will be authorized in some specific low-load hours of the day despite their SLA is defined during working hours only. Note that policy agents can be created or modified for specific dedicated applications, as described in ONAP within the design framework portal and in policy-based middleware MASC [22] architectures. These agents are then kept in a policy repository for distribution to relevant orchestrators.

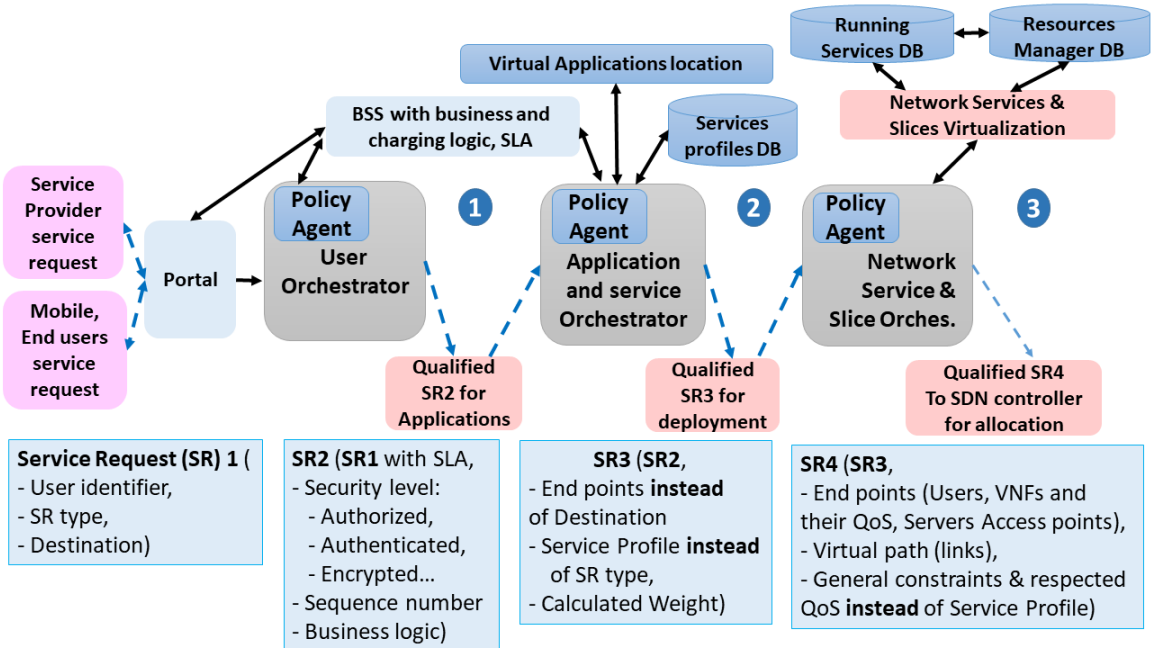


Figure 8: Organizational proposal at the upper layers

Once qualified, the SR includes security level, service type, and destination. It is directed to the application and service orchestrator.

(A) First, we describe **an SP network slice request** that illustrates the setup phase of a zero-touch network. SP signs an SLA with an operator for a network slice with defined QoS and the connection points to the operators' network. The second orchestrator associates the endpoints and their network location and looks for the corresponding service type in the service profile (catalog) database (DB). The request includes QoS parameters such as minimum throughput, maximum delay, and error ratio [23]; list of access points; security level; protection type; and functions associated to the service. The SP request is then addressed to the policy agent that includes decision-making rules related to business logic or regulation. With the help of the BSS and the policy agent, a service weight is calculated according to the customer importance, the signed SLA, and the SR priority, e.g., best effort, guaranteed, or high priority. This can be enriched according to the complexity of the network and the services and user's types. High weight can be reserved for public safety services such as firefighters. This is critical in case of severe network failure, for instance, when some running services must be removed. Instead of removing services blindly or randomly, the weight can be checked at the resource orchestrator. Lower weight services will be removed first, without needing to re-qualify the services at a higher level of network management. Instead of service type request, with the help of the service profile DB, there are now general QoS parameters and services constraints. The qualified SR is then directed to the NSS orchestrator that calculates the network slice with the help of NSS virtualization (see Figure 8, step 2). The network slice is attached with its components: the VNFs with their properties and constraints (located in the resource manager catalog) and the links of the slice. This second orchestrator is aware of the service applications. For instance, when there is an end-user request to

a SP video streaming server, this orchestrator knows the different locations in the network of these application servers (This can be done with the help of a DNS application server.). According to network or application load considerations, it may recommend one server rather than another.

In step 3, the service is virtually deployed, and this request is sent to the general SDN controller for path calculation with respect to QoS at the resource level. If the resources are free, the network slice is physically allocated. The network resource orchestrator works with the main SDN controller and synchronizes resource applications such as self-healing, energy saving, or path optimization running in background. Technology orchestrator operates per technology, at the level of the different SDN controllers. It can optimize optical path or equipment usage such as antenna power or can shutdown unused equipment in order to save energy. The running services and resource manager DBs, the performance monitoring (PM) handler, and the resource orchestrator are updated. The PM handler allows real-time monitoring of the running slices and of the resources: there is a NOS per allocated service. It can update the orchestrators or the OSS if required.

(B) When dynamic changes occur in the network slice, several components may have to be modified, e.g., at a large musical evening event (The SP charging policy must be considered here.). The autonomous network should be able to renegotiate the existing slice with attachment of alternative resources or temporary additional capacity, first at the NSS virtualization layer. Some access points located near the musical event require more throughput, and some VNFs may be migrated or added to other VMs. The NSS orchestrator attaches the virtual components. Then, the SDN controller and the lower layers check and physically allocate the attached slice. The slice is then physically modified. The databases are updated for the period of the event. Using two virtualization layers enables work in parallel and assures network elasticity and dynamicity. The virtual deployment is not limited by the network resource virtualization, i.e., the network slice can be created without every limitation of the resource's virtualization.

For example, when a network slice including virtual base stations (vBS) is modified, these VNFs do not always exist in the required VM (closer to the event). They can be scaled up or created at the step of the network slice allocation. Open-source software such as Docker and Kubernetes that support containers and containers management, cover this domain with micro-services (used in CORD and ONAP) (see section 2.3). Consequently, this separation into two levels of virtualization in the NaaS assures dynamicity of network slicing. Moreover, network slices stay isolated from each other. In this case of vBS overbooking scenario, as described in Figure 9: vBS scenario, a set of antennas in the event place is connected via a distributed antenna system (DAS) to the network (This can be based on a set of mobile antennas specially deployed for the event.). The operator can migrate up to **25 vBS** on VMs in datacenter near the musical event. For the event in the evening, three mobile operators require resources: Cellcom, Partner, and Pelephone. As a choice, each company needs **8 vBS** for the evening to support their mobile subscribers. During the evening, Cellcom and Partner operators need another **vBS** to cover their needs and every operator has already received its vBS resources. So, with this additional mobile resource request, 26 vBS are required for the event and one vBS is missing. According to the chosen SLA in this scenario, Cellcom or Partner operators are equivalent from SLA point of view. If we suppose, a vBS cannot

be attributed to two operators simultaneously, then only one operator will be able to receive the last vBS. In case of autonomous network, how the orchestrator will select the operator that will receive this last vBS? If we want to work according to the rules driven decisions process, the operators will have to be re-qualified and get new weight according to additional rules in order to affect the last vBS in a rational way and maybe loose some SRs up to a correct decision. Regarding this overbooked situation, the decision for vBS attribution may be also done randomly to answer the real-time needs of the customers and bring the network to a more efficient behavior where at least part of the customers receives a real-time answer to their needs. Simultaneously, re-qualification process can be done in background and bring to more rational decision in order to optimize the general network performances.

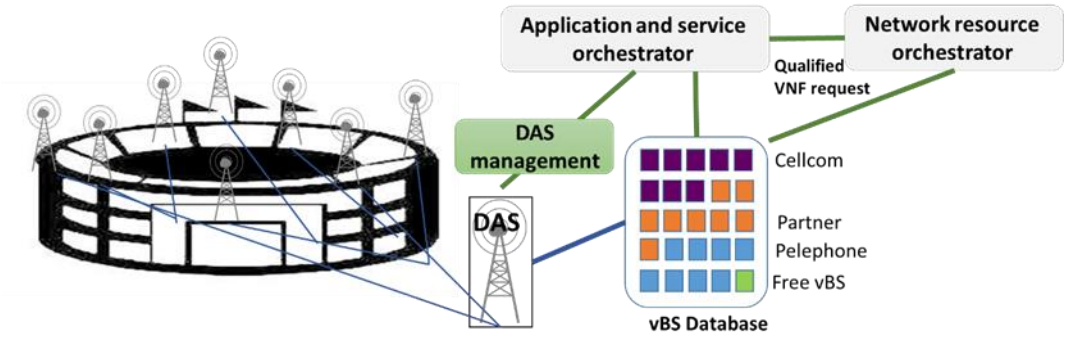


Figure 9: vBS scenario

In case of network failure (e.g., fiber cut or link down), the network slice is modified differently according to its protection type:

1. The network slice may switch automatically to its defined slice protection (if a physical equipment protection is defined, reserved, and included in the "Data plane SLA"). This is a protection at the **physical level**.
2. The switch to protection may be also at the **network resource level**. The path of the slice is recalculated in the SDN controller with the new limitation, and the slice is updated with the new links and nodes in agreement with "Control plane SLA".
3. Another possibility is to operate at the **virtual network level**. Some alarms and PM are sent to the orchestrators and slice DB. The Network slice is re-qualified and renegotiated with NaaS SLA: it requires modification of the components with the NSS orchestrator, and the same slice updating process is performed and the running services DB is updated. Note that in this case, the existing slice can go on running in degraded mode until there is a switch to the new or updated slice. This way, we can avoid a service interruption and thus grant more dynamicity to the network.
4. If the latter is not enough and the link to a SP server is damaged for instance, according to PM monitoring, the slice can be re-negotiated at the **application and service level**, where another available SP server can be selected. The network slice is then attached again with other links and nodes and switched and deployed with the help of the general SDN controller.

3.4. Architectural and organizational proposal conclusion

The growing complexity of networks and the explosive demand for services drive the need for a zero-touch network. The orchestration is one central piece to create such a network. Among the different management layers, an architecture that overcomes limitations in the area of orchestration at the northbound interface with the OSS is required. Synergies between the SDOs and open sources would help this to be achieved. Although these groups have performed substantial work in the domain of standardization and adaptability between the control and data planes, these architectures lack some dynamicity. In this chapter, we proposed an enhanced architecture by introducing a new network virtual layer above the SDN controller to improve dynamicity. With this new layer in place, orchestration is then distributed among different layers. We clearly identified the role of orchestration and the resulting greater autonomy. We showed how this architecture can overcome some limitations such as simultaneously respecting the SLA and QoS constraints at different levels of the network. By using an organizational model, we explained how this architecture provides more autonomy and more dynamicity when a slice, a VNF, or some resources must be modified, migrated, or added during a session. Following this organizational analysis, in Chapter 4, we are interested in a functional approach where sequence diagrams better identify the different modules and their interactions. Using a simulation carried out in Chapter 5, we also demonstrate the gains of such an architecture to illustrate the validity of our proposal.

Chapter 4. Functional proposal: sequence diagrams and scenarios

In this chapter, we illustrate the functional aspects by using sequence diagrams to describe different scenarios of service provisioning and modification to gain clear understanding of the autonomic and dynamic aspects of our architecture. The studied cases are the followings:

1. N simultaneous services requests,
2. Dynamic service extension during user's session,
3. Overloaded link,
4. Voice call that needs to be encrypted in the middle of the conversation.
5. Energy optimization

Each case is studied in a different section and illustrates the proposed architecture from another angle. The case of multi-operators and multi-domains cooperation and interoperability to create a service is described in MEF 55.1 [24]. Interesting use cases are described there where the role of the Cantata API and its corresponding support functions are clearly identified with the help of sequence diagrams.

4.1. N simultaneous services requests

The Figure 10: Common case sequence diagram for service allocation refers to the **first case** once n SRs are already attached as virtual services (see previous chapter) independently of the layers below. This describes a common scenario that does not require any specific dynamicity or autonomy, at least at the beginning. The aim is to show the function of service/slice complete creation and network monitoring considering the local QoS together with the E2E service constraints with the help of the service and slice orchestrator. There is a request to the SDN controller to calculate the path and the required VNFs locations according to local QoS constraints at the resource level. The request is then sent to network resource virtualization, where the choice of the virtual nodes, Vlinks, and VNFs is performed. For example, a virtual CPE can have to be located at a limited distance of a VM running an application related to this vCPE. If the resources are available, confirmation is sent to the SDN controller, and the service placement is performed on the physical network where the service is activated. Finally, the running services DB is updated with the new services together with the PM and statistics handler, and virtual NOS (VNOS) [10] monitoring starts on this link. VNOS per running service once activated, allows continuous and autonomic service management, and can detect if the QoS is not respected (use cases explained in next sections).

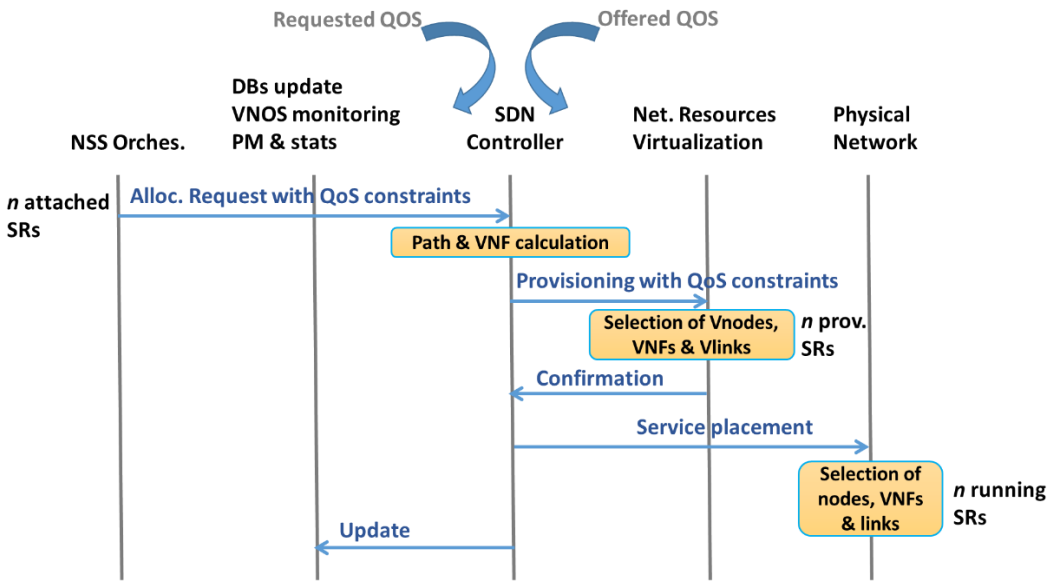


Figure 10: Common case sequence diagram for service allocation

4.2. Dynamic service extension during user's session

In the second case, we describe how the proposed architecture behaves in case of an overloaded VNF. We want to show how the processing resources of a running VNF can be dynamically modified during a session to answer the service request changes. We suppose CPU usage is one of the major metrics of the VNF located on a VM, and it is measured continuously. In case an active running service uses more than 90% of a VNF computing resource (see Figure 11), a VNF extension is required. VNOS service monitoring is updated, and the service is sent for re-qualification to the resource orchestrator. Rules are defined in the policy agent of this orchestrator, e.g., "if more than 90% of the CPU is allocated for a VNF, allocate another 20% CPU for this VNF". This resource extension is requested to the main SDN controller, which asks for 20% more CPU to the VM where this VNF is running. If there is enough free resource on the VM, confirmation is sent to the SDN controller, the service extension is placed, and the running service is scaled up. At the same time, the running service DB and the resource manager are updated with the new VNF, VNOS monitoring of this updated service is going on. In case there is not enough free CPU on the VM, the service can be re-qualified and sent to the third orchestrator to find another available VM respecting the SLA and QoS constraints. The updated service is activated when deployed instead of the previous one, and thus avoids service interruption. In ONAP or CORD based on new SDO architectures, the control loop should allow this dynamicity if there are enough resources on the VM. However, if the VNF must be moved to another VM, we will likely not be able to avoid service interruption or severe degradation. We understand in this case, the importance of an intelligent information that allows decision making, of the continuous monitoring with the VNOS and of practical and efficient rules definition at the level of the NSS orchestrator. This VNOS-based monitoring service can enable new proactive management functionality. If the resources of a VNF of this service are overloaded, the maximum threshold of

other components of this service may be about to be exceeded. Thus, a test on the load of the various components of this service can be operated proactively, in order to operate preventive actions which will allow a resizing (scale-up) of the current service and thus avoid other events of this type.

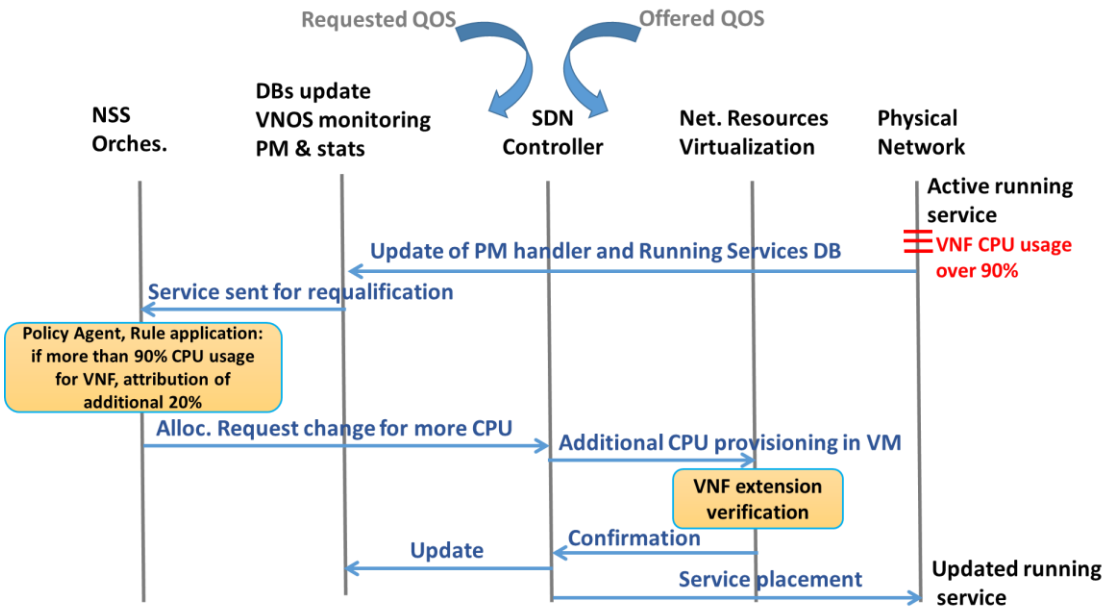


Figure 11: Service dynamic modification with VNF resources requests extension

4.3. Overloaded link

In the third case, we study the functional behavior of the architecture in a case of overloaded link. Although similar to the previous case, here the network resource orchestrator function with its rules is mainly involved. When n active services are running through specified Link1 that is used at more than 90% of its capacity, (see Figure 12), this link status becomes overloaded (depending on policy operator). The running service DB and the PM handler are updated, and service re-qualification is sent to the resource orchestrator this time. The defined rule can be for instance: "In case of overloaded link (more than 90% usage), remove the lower priority services going through this link until getting back to 80% link usage and re-qualify the services that have to be removed with a new path calculation where Link1 is forbidden". Re-allocation of low priority services is sent to the SDN controller with the new constraints. The SDN controller calculates the new path, which is then treated in the usual way. Here, the offered QoS has satisfy the required QoS. If there is an alternative free path, virtual resources are reserved and then allocated in the physical network. Note that in this case, the original service is removed only when the new service is ready. Thus, the switch to the new path is transparent to the end-user and assures better QoS and enhanced network dynamicity. Other rules can be applicable in the network resource orchestrator such as "forbidden link usage until link usage comes back to 80%". If this link is often overloaded, a longer-term event may be generated indicating that this specific link

needs to be upgraded with more capacity. In ONAP or CORD based on new SDO architectures, the control loop should allow this dynamicity if enough resources can be freed. However, we will likely not be able to avoid service interruption or severe degradation. This case is simpler and can be treated at the level of the network resource orchestrator. There is no need to associate here the NSS orchestrator or VNFs manager.

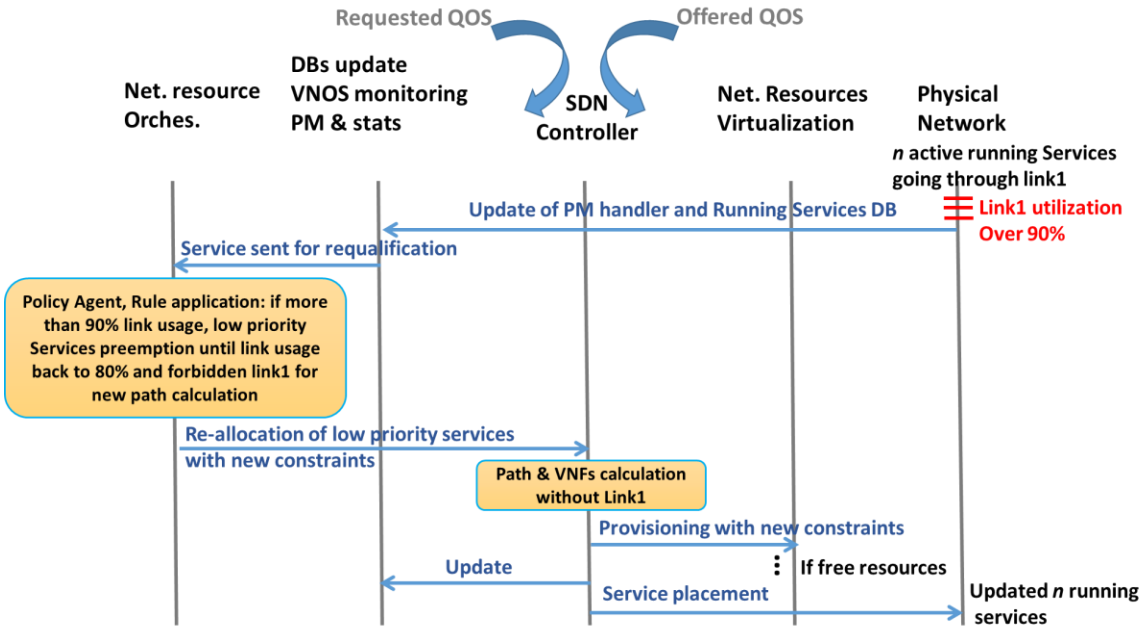


Figure 12: Dynamic service modification when a link is overloaded

In case of an overloaded direct link to one content server as described in section 3.2.2, in today's networks the service is progressively lost; it is first degraded with less capacity until it cannot be delivered anymore. In our architecture, when access to the application content server crosses a defined threshold, the PM handler sends an alarm, and the lower priority running services have to be re-qualified. (Saturated application content server is beyond the scope of this paper.) Considering that the application server is unavailable, the service is re-qualified at the application orchestrator and recalculated. If another available equivalent application server is found, this service is recalculated and redeployed with another path and other VNFs, and the running services are switched to the new one. This service switch can be transparent, and here also the QoS is preserved. Video streaming on YouTube or Netflix applications, is protected at the level of the application in order to find another server (often further), where the video streaming can go on. There is no decision at the level of the network function itself.

4.4. Modified phone call in the middle of the conversation

In the fourth case, we return to the example of the phone call described in section 3.2.2 where we want to encrypt a voice call in the middle of a conversation. The originality here lies in the fact that there is a function of two VNFs in the middle of a session. Usually, once the SRs are

attached, they are sent to the SDN controller to calculate the path and the required VNF locations according to the delay and QoS constraints at the resource level. Then, the request is sent to network resource virtualization, where virtual nodes and vlinks are chosen. If the resources are available, confirmation is sent to the SDN controller, the service is placed and activated. Finally, the two databases and PM handler are updated with the new service, and VNOS monitoring starts on this link. In this call scenario, a request is sent to modify the running service. This request is addressed to the NSS orchestrator (see Figure 13). It looks for two VNFs realizing encryptions in the resource manager, to allow this E2E call encryption. To this end, these two VNFs must be deployed near the end-users to comply with an E2E encrypted call. The orchestrator tries to recompose the service at the NSS virtualization. They require more VM computing resources and introduce queuing. Together with the other VNFs of the call, this may overcome the E2E QoS call constraints. A request to the NSS orchestrator can improve the E2E delay by decreasing, for instance, the maximum number of hops to set up the conversation. Then, once the SR is attached, it is addressed to the general SDN controller that calculates its path, using the network resource virtualization. If these VNFs already exist in the related VM, they are activated, the running service is updated or re-allocated, and the service is switched transparently to the modified one. If these two VNFs do not exist in the needed VMs, one option is to install these VNFs in the required VMs (If enough CPU and memory) with the help of micro-services. Else, some migration of other VNFs can be done before, if these VNFs can be located in another VM and thus free resources for the required VMs. Then, the databases are updated and the VNOS associated with this service restarts. In today networks, such a dynamic request is not handled: the call must be stopped, and if encrypted call service is available, the call service must be renewed. In the ONAP architecture, for instance, one of the most advanced open sources, the complete service instantiation uses CLAMP, the Closed Loop Automation Management Platform [17]. It is used to design a closed loop, configure it with specific parameters for a particular network service, then deploying it. Once deployed, the user can also update the loop with new parameters during runtime, as well as suspending and restarting it. This is done with the help of interaction with the monitoring, with function catalog and the policies of the orchestrator that is also updated by the control loop. It can be represented as a continuous workflow until the closed loop is un-deployed. However, the virtual functions and the virtual networks are not at the same layer as described in section 3.3.3. There is no clear separation of network and resources layers with their associated orchestrators. The approach stays network centric.

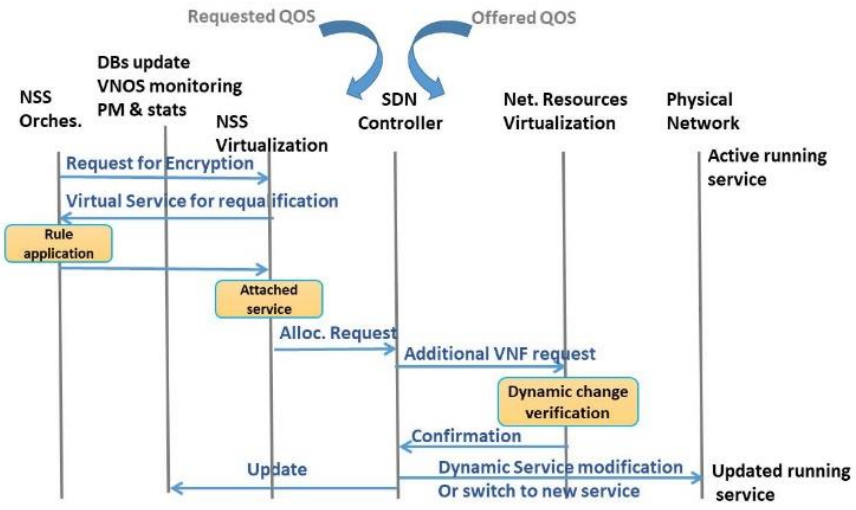


Figure 13: Request for call encryption during a conversation

4.5. Energy optimization with ETSI

The ETSI case is interesting because of the use of AI to solve specific scenarios. In ETSI GS ENI 001 [25], several use cases are described to improve the network efficiency with the help of an ENI (Experiential Networked Intelligence) module. The described use case proposes to optimize the use of energy at the level of the servers and VMs of the data center, which represent 70% of the energy consumption. This is made possible by the use of techniques such as AI used in the ENI and VMs/containers, which allow the scale in/out or the dynamic migrations of VNFs.

This case can be adapted to our architecture in case an AI function is integrated in the NSS orchestrator together with the continuous monitoring. Our case is also different from the ETSI case because the physical machines where the VNF are running, are distributed over the network and the migration of a VNF should be verified with the NSS to be sure that the local and E2E constraints of the running services are still respected. In our case, the AI module and its rules are located at the NSS orchestrator at the corresponding layer, instead of the ETSI ENI module. If the constraints are respected and if the still running services on the candidate servers to be switched off, can be moved to another physical machine, these physical machines will be switched off according to the services usage pattern (see Figure 14). This services usage pattern is created with the help of the continuous monitoring and the AI module learning from the network usage. The collected monitoring constitutes the data, there is an offline training for the machine learning to create the model and usage pattern.

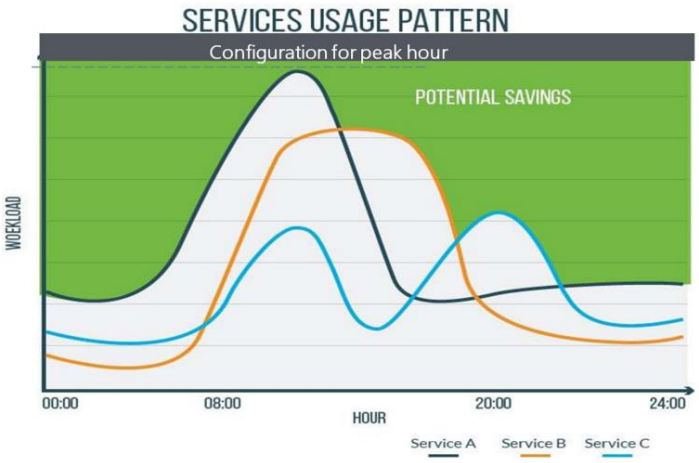


Figure 14: Potential physical machines energy saving with the NSS orchestrator[1]

The sequence diagram is described in Figure 15. The monitoring data is also collected by the SDN controller, the NSS virtualization database, where every virtualized and allocated services are kept and managed by the VNOS. When a potential saving period is identified, according to the built model, a command is sent to the NSS orchestrator to switch off several physical servers (PM) during this silent period. Before sending the command to the SDN controllers for execution, the command is verified at the NSS virtualization DB to see if services are still running on these servers. If a service is found on one of the chosen servers, the virtualized service is modified with the new constraint (not to use this PM). Then, the updated virtualized service is sent for allocation to see if the local and E2E constraints can be still respected using another active PM. If yes, the VNF of this service are moved to the new physical machine (the usage of containers should allow the VNF modification) and the DB is updated. If not, the switch is not possible, and the corresponding server cannot be set to sleep or shutdown. The command from the NSS orchestrator can be sent every few minutes during the silent period as a background activity, to try again to switch off the server. Monitoring is permanent and continuously updates the database of running services.

[1] See figure in ETSI GS ENI 001 specifications

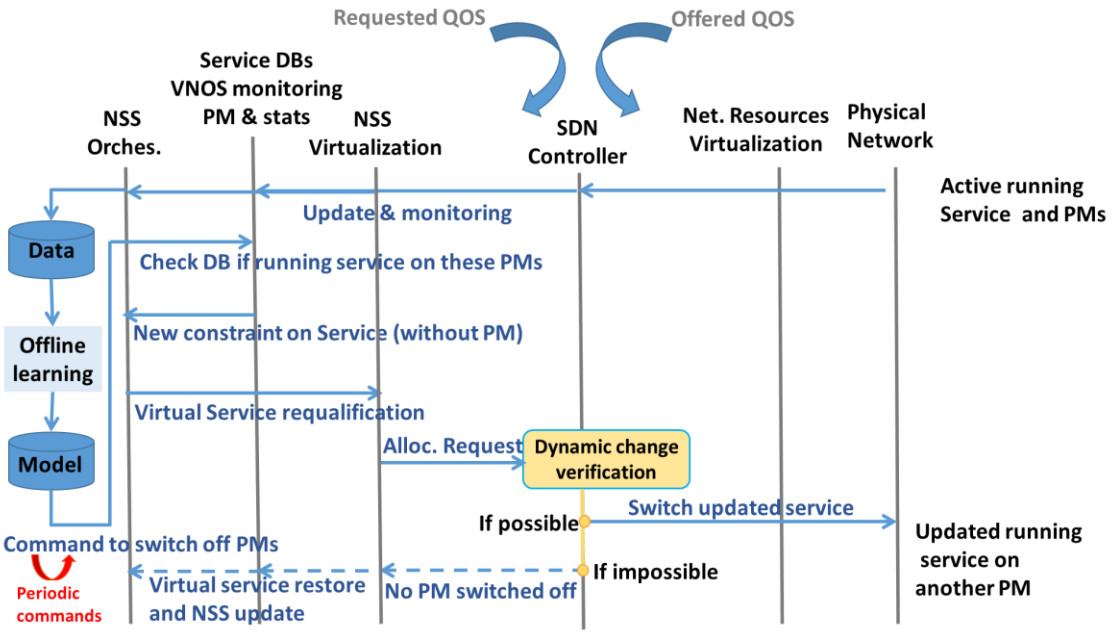


Figure 15: Request for virtual machines energy optimization

This scenario shows the interest of the different orchestrators and the virtual network layer. It strengthens the network's autonomy and dynamicity. it also illustrates the potential of machine learning function at the orchestrator level.

4.6. Conclusion of the functional proposal

In this chapter, a functional analysis, use cases, and sequence diagrams, help us to better understand and identify the advantages of using the new elements of architecture described in Chapter 3. Through the above scenarios, we have identified several functions: the need for different orchestrators, particularly at the levels of the network services and slices and network resource; these orchestrators need many rules to be able to respond to the maximum number of scenarios; the function of continuous monitoring with the VNOS to be able to modify dynamically and even proactively a service if necessary; the decisional feedback function which requires a notification bus where the events have to be distributed to all relevant functions for intelligent qualification; the AI function which can be helpful where some network optimization is performed for instance. We have seen that given specific and different scenarios, our architecture only can warranty the autonomy and dynamicity during a session, of the network operation together with the respect of the E2E and local QoS constraints. In the next chapter, we demonstrate with the help of a simulation based on open sources, the validity of our proposal.

Chapter 5. Architecture simulation

The aim of this chapter is to demonstrate the interest of the proposed network management architecture and its organizational and functional models with the help of a simulation, particularly in the domain of the service dynamicity during a session (see Chapter 3 and Chapter 4). This example also illustrates the resulted enhanced autonomy.

5.1. Chosen architecture and aim of the simulation

The Simulation is based on the architecture described in chapters 3 and 4 (see Figure 6). Because of its complexity and its many modules, in this chapter, we simulate only part of this architecture, the goal being to demonstrate its dynamicity and its autonomy. Also, we choose a scenario which illustrates these properties, and we only implement the necessary modules for this demonstration.

5.2. Methodology

The methodology used to show that this architecture can respond to dynamic modifications of services (on demand) when more traditional architectures require a re-creation of the service, is carried out using a simulation. This simulation will take place in 3 stages:

1. The first step is to show what is possible with existing architectures, including what dynamics and autonomy can be achieved.
2. The objective of the second step is to demonstrate the limits of current architectures and to illustrate with one or two examples what is impossible to do.
3. The last step consists in showing the improvement of the dynamicity brought by the use of a specific orchestrator corresponding to the involved layer and by adding the virtual network layer.

5.3. Platform and simulation and performance tools

A COTS server (X86 Intel based) is used with one CPU of 8 core and 32 GB RAM as platform for our simulation. To simulate a network, we use the following open sources:

1. Mininet is a network emulator which creates a network of virtual hosts, switches, controllers, and links. Mininet hosts run standard Linux network software, and its switches support OpenFlow for highly flexible custom routing and Software-Defined Networking. Mininet runs on Linux of Ubuntu. Mininet can easily simulate multiple topologies by modifying a python topology file used to launch Mininet. It also allows to introduce errors in the different links.
2. Open Network Operating System (ONOS®) as mentioned in Chapter 9, is also installed in the same virtual machine as Mininet. It is one of the main open-source SDN controller from ONF forum to implement next-generation SDN/NFV solutions. ONOS can work together with Mininet by automatic discovery of the network topology implemented with Mininet.
3. Wireshark is also used to measure the performances of the applications together with tools like "iperf" that are available on Mininet.

Mininet, ONOS, and Wireshark are installed on the same virtual machine.

5.4. Performed simulations

5.4.1. First simulation: capabilities of existing architectures

In this scenario, we want to show the limitations of existing architecture. For this purpose, we overload the network using ONOS and Mininet. The figure below presents the topology used for this simulation. This is the graphical user interface of ONOS. This network is composed of 6 switches and 18 hosts.

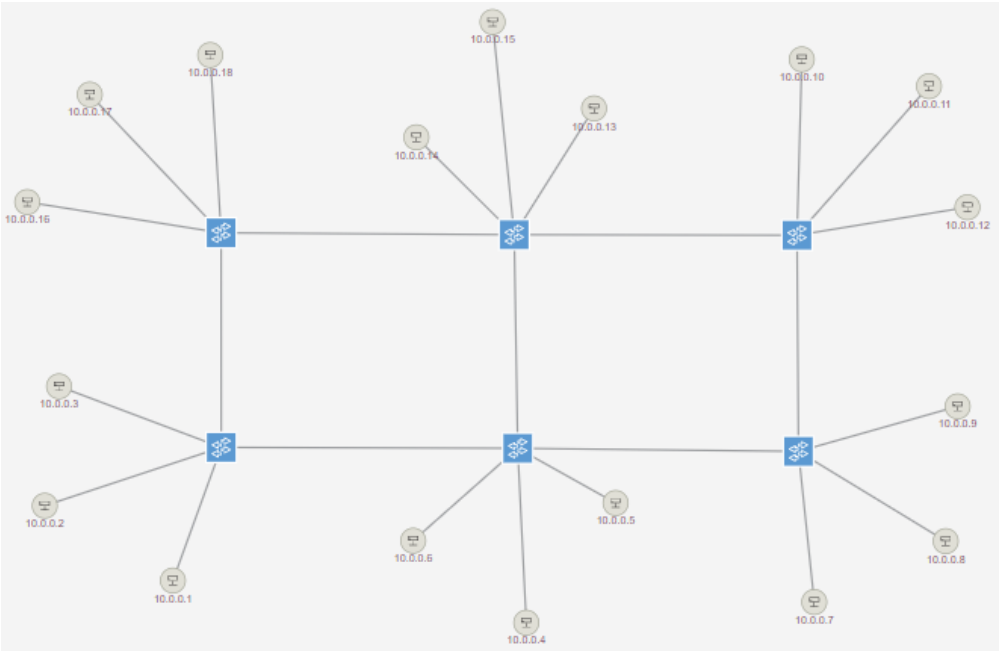


Figure 16: Overload scenario displayed in ONOS

In this case, we use iperf tool between host 10.0.0.1 (host1) and 10.0.0.11 (host11). This tool can generate traffic between hosts according to the network capacity. This traffic can be monitored. We configure host11 as a UDP server and host1 as a UDP client:

On xterm host11, we use the Mininet command: iperf -s -u -i 1 # with "-s" for server, "-u" for UDP, and "-i 1" for monitoring every second.

On xterm host1, we use the Mininet command: iperf -c 10.0.0.11 -u # with "-c 10.0.0.11 -u" for UDP client transmission to host11 UDP server.

In this case, the results are correct and there are no errors.

In the second case, as detailed in Figure 17,we simulate errors in the links between the hosts and the switches and between the six switches: we use a low bandwidth (bw=1 Mbps), add a 5ms delay and a 10% loss, and limit the size of the queue. The host11 is running the same UDP server with the same monitoring period.

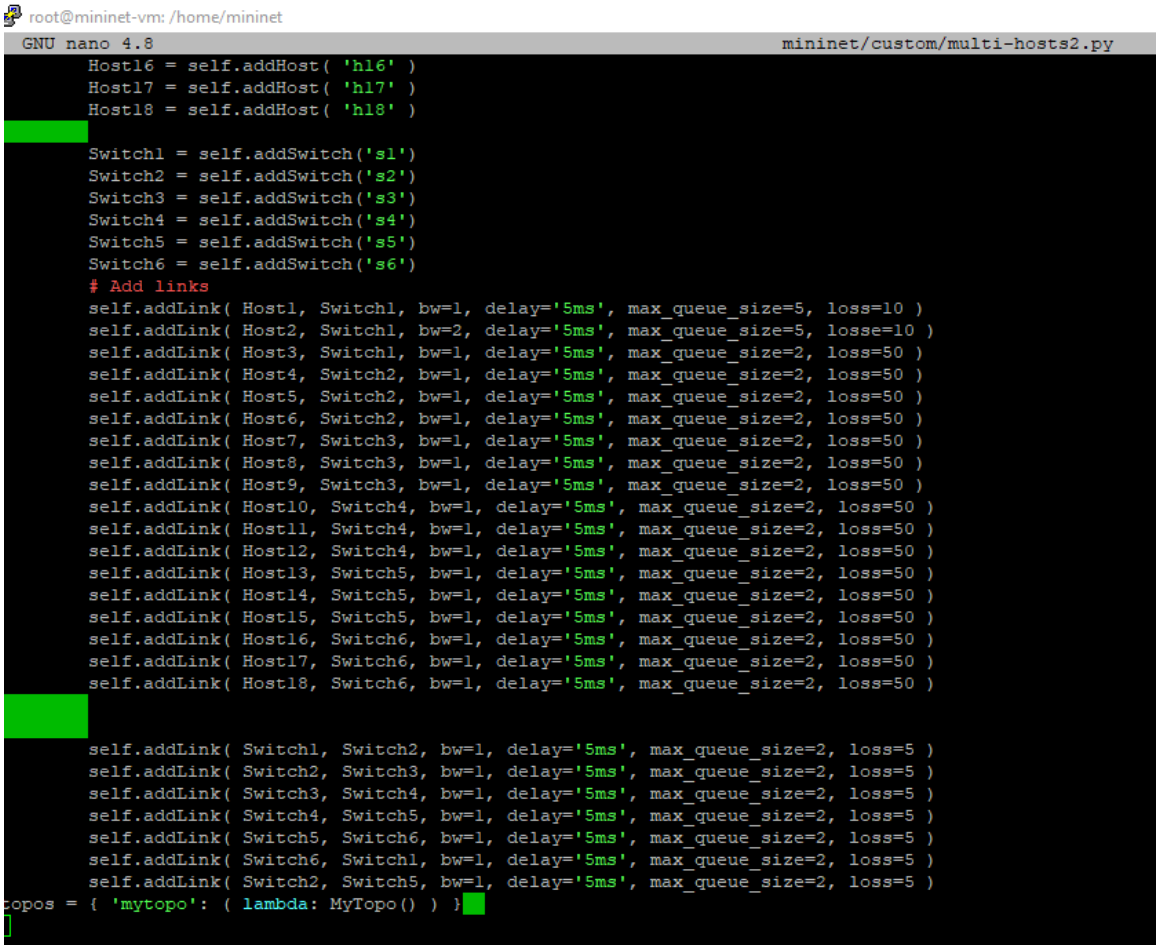


Figure 17: Modified topology using python

However, on xterm host1, we now use the following Mininet command: iperf -c 10.0.0.11 -u -b 1000M # where the traffic is modified to a bandwidth of 1Gbps for UDP client transmission to host11 UDP server when the capacity of the links in this network is about 1Mbps.

In this case, as displayed in Figure 18, we can observe the very limited bandwidth, the variable jitter, and the very important percentage of lost packets.

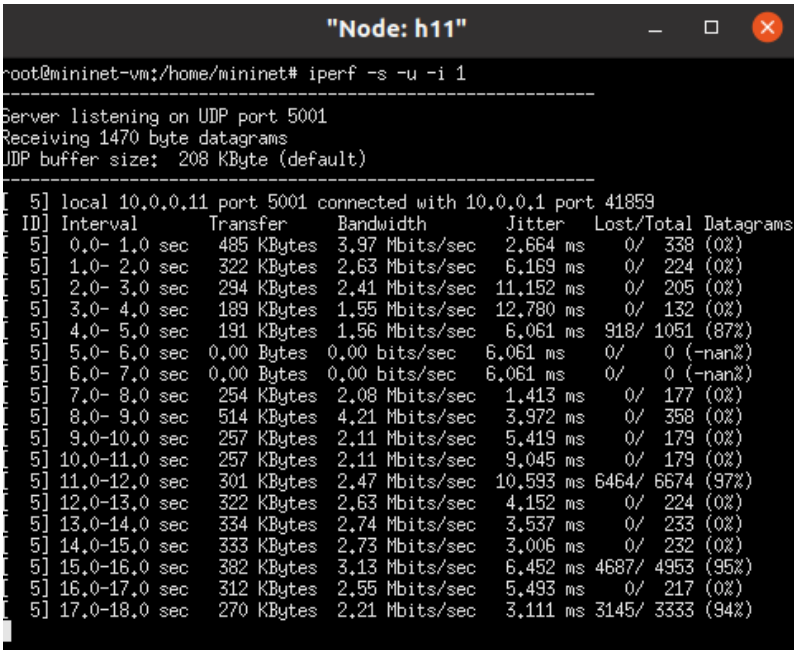


Figure 18: results of the simulation when the network is overloaded

With an SDN controller such as ONOS, traffic can be scaled based on changes in link properties. However, when the flow is too high, the network can no longer cope, and the percentage of loss depends on the overflow scenario. Thus, the current networks can adapt to increasing load and show some dynamicity, up to a certain loading point where the network collapses and can no longer cope with the overload on its own.

We continue our simulation to show the contribution of orchestration and a virtual network layer to adapt to dynamic changes during session and maintain the network autonomy.

5.4.2. Second simulation: dynamic video service

As described in section 3.2.2, the second simulation is based on the knowledge of the orchestrator and on the monitoring of the virtual network layer. In the topology described in Figure 19, we have the same videos on two web sites located on h2 and h3, and one host with web client wants to obtain a specific video to one of the nearest sites. The last link to the websites has 10 Mbps to make the video download last. We implement a limited search engine h4, that proposes these two sites when searching for one specific video. The web client receives two IP addresses of these web sites. A simple ping test to the WEB sites can be used to select the nearest server. While the client is downloading the video, some errors on the links (or link down) are introduced until the

service is lost. In this case, the service cannot be reconnected automatically to the second web server with the same film except in the case of Netflix or YouTube, where there are some redundancy mechanisms at the level of the application.

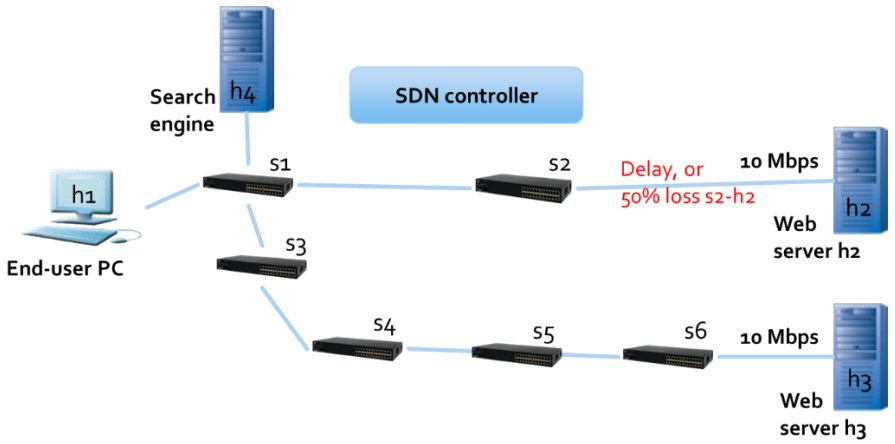


Figure 19: Web sites with video servers with a SDN controller

In the second case (see Figure 20), we use the same scenario, but we add a virtual network layer and an orchestrator. When the service is created between h1 and h2, it is also created and monitored on a virtual layer. When the link to h2 degrades, the created service becomes aware of the problem. But there is no other path to h2. So, the service cannot be rerouted at the network and SDN controller level, and the network resource orchestrator cannot help. Considering the link to the application server is affected, the service has to be recomposed with another equivalent video server. An event is sent to the service monitoring (E2E slice) that updates the orchestration. Because the video server h2 is unreachable, the application orchestrator is invoked and looks for another server that will be able to deliver the same film. The application orchestrator addresses its request to the search engine h4, that proposes the same two http video servers (Theoretically, the application orchestrator should maintain a table of the available video servers with their respective cost, so the request to h4 should be useless). Considering the server h2 is unreachable (after ping), the server h3 is selected and the service is recalculated with h3 as the second endpoint. The service is virtually created and the SDN controller calculates the shortest path to h3. Once h3 server is selected, the http server can download again the same film. The switch to the new service is dynamic (during session) and automatic (without human intervention) and is operated at the level of the network management.

Note: the scripts and functions on the different servers h1 to h5 were written in Python and Wireshark is used to record the results and the performances of the simulation.

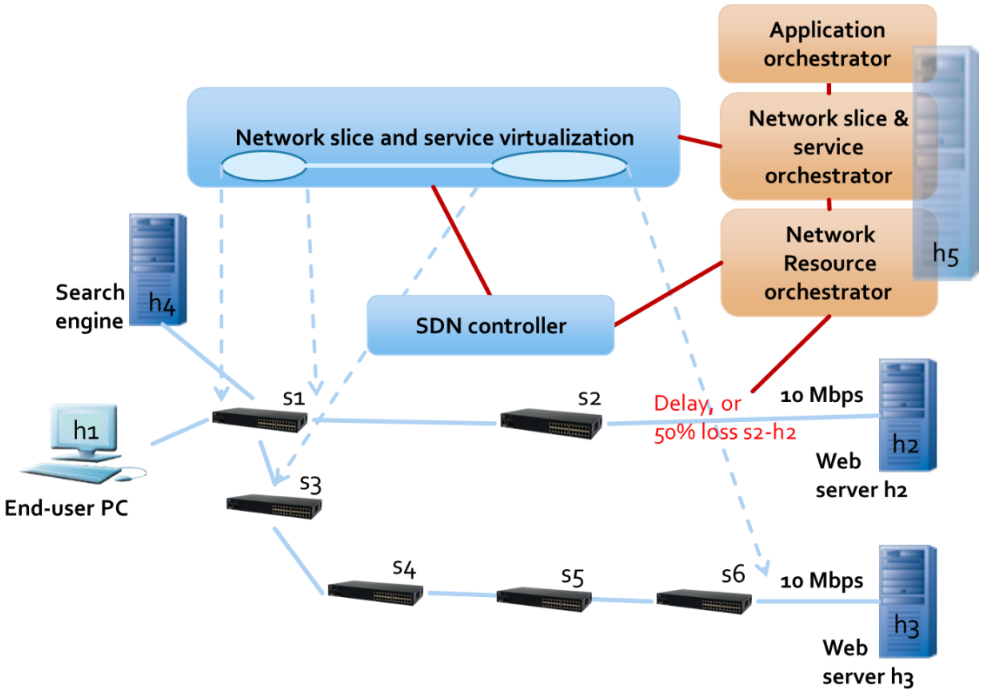


Figure 20: Web sites with video servers with an orchestration and a virtual layer

5.5. Result analysis

5.5.1. Service interruption

As displayed in the Linux screen (see Figure 21), our Mininet topology is running (bottom right side of the screen). The two video servers, the search engine, and the orchestrator h5 are activated. The client h1 asked for a specific film to the search engine h4. H4 pings the two servers h2 and h3 to see what the nearest server is. H1 receives the IP address of h2 and asks for a specific video.

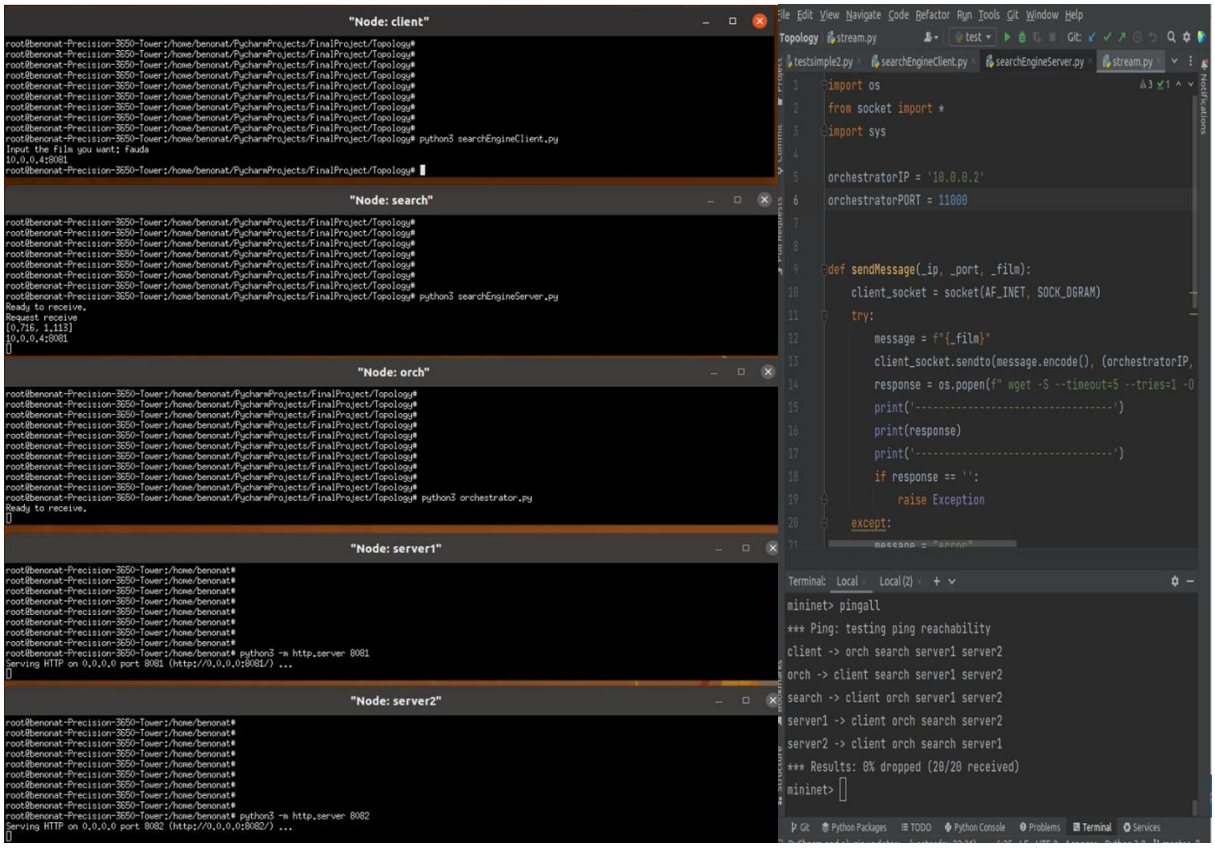


Figure 21: Video request to server h2

When the link s2-server h2 is degraded, the download is interrupted on h1 and an error event is generated in Mininet, an alarm is sent to the orchestrator h5. As displayed in the Figure 22, the link s2-h2 is down, the download is interrupted at 27%. The orchestrator receives an error message (see Figure 23). Considering there is no alternative path to server h2 in this topology, the application orchestrator looks for another server with the same video using the search engine. The ping to h2 and h3 is sent again from the search engine but now, h2 is unreachable. So h3 server IP address is sent to the orchestrator that allows the automatic modification of the service that connects now to h3 instead of h2 (see Figure 23). In this figure, 16% of the h3 video is down. The search terminal is indicating the server h3 whose address is 10.0.0.5 and port 8082.

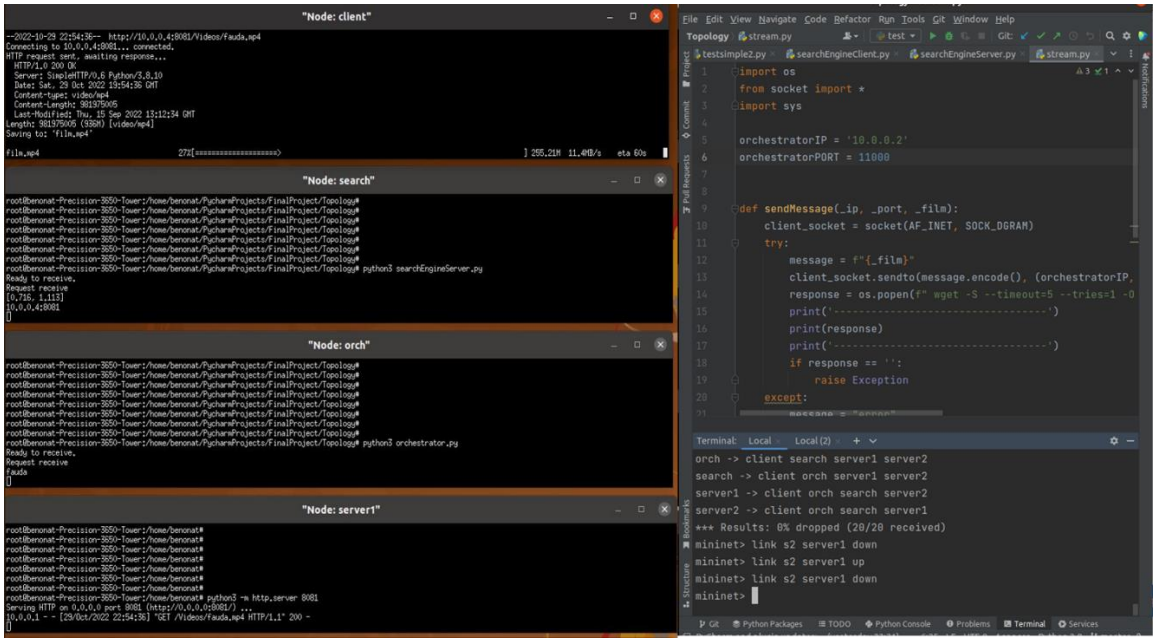


Figure 22: Video download interruption when link s2 server 1 h2 is down

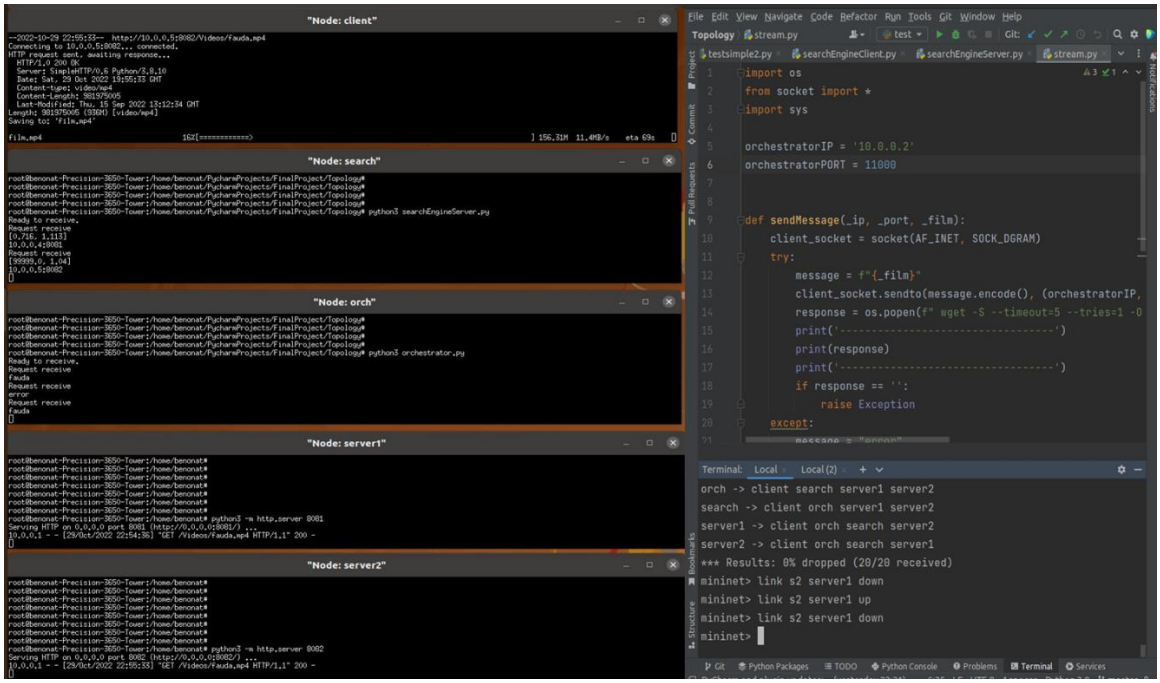


Figure 23: Video download to server h3

After detection of the error on the link s2-h2 (server1 in the screen), the switch to the new server is automatic and a dynamic change occurs during a service session. The download is not going on but is reset because we consider the video content service providers are different. This video can exist in two different content service providers.

5.5.2. Performances

We can follow with the help of Wireshark the different steps of the previous simulation. The Figure 24 shows the error that is sent from the client to the orchestrator 10.0.0.2 as a UDP event. Also, we can see an LLC layer two message from the orchestrator to the search engine 10.0.0.3 to look for another server for this specific video.

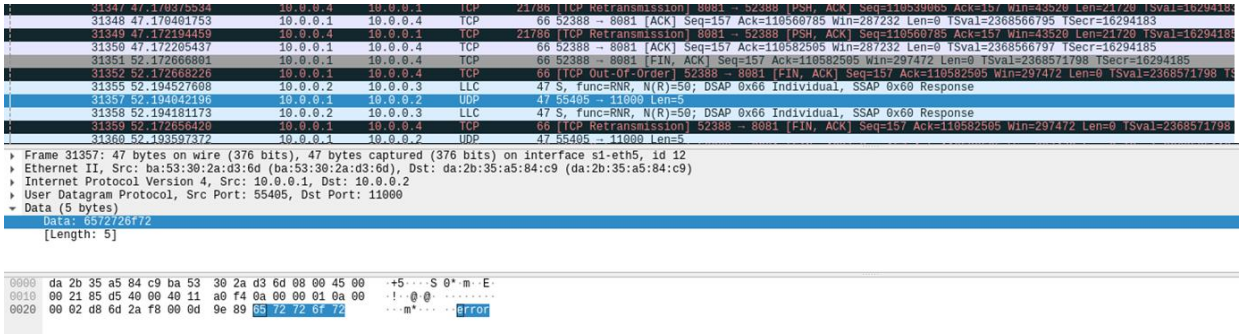


Figure 24: Wireshark UDP error message to the orchestrator and request to the search engine

In the Figure 25, the HTTP request from the client to the second video server 10.0.0.5 is displayed. The TCP setup synchronization between the two servers is also recorded.

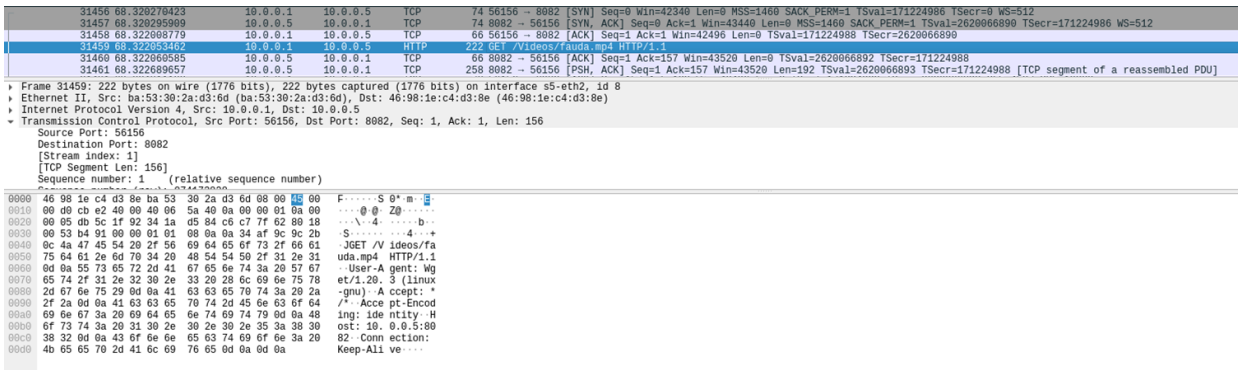
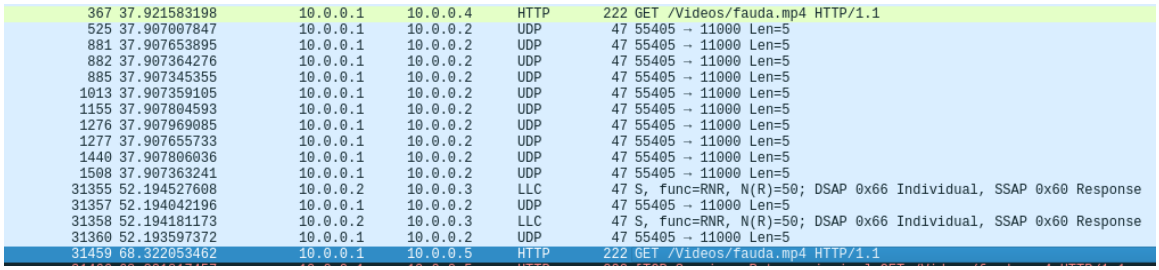


Figure 25: Request to the second video server

In the Figure 26, the switch between the first and the second server is measured. The complete switch lasted 30.4 seconds (between 68.32 and 37.92). This is mostly due to a network solution, to a retry connection to h2 that is performed at the client level (part of the script) and the ping function (4 echo ICMP requests) for each server, that is realized at the search engine before choosing the second server. As explained earlier, if the application orchestrator maintains a table of the available video servers with their cost, this time will be shorter and will not take more than a few seconds.



367	37.921583198	10.0.0.1	10.0.0.4	HTTP	222 GET /Videos/fauda.mp4 HTTP/1.1
525	37.907007847	10.0.0.1	10.0.0.2	UDP	47 55405 → 11000 Len=5
881	37.907653895	10.0.0.1	10.0.0.2	UDP	47 55405 → 11000 Len=5
882	37.907364276	10.0.0.1	10.0.0.2	UDP	47 55405 → 11000 Len=5
885	37.907345355	10.0.0.1	10.0.0.2	UDP	47 55405 → 11000 Len=5
1013	37.907359105	10.0.0.1	10.0.0.2	UDP	47 55405 → 11000 Len=5
1155	37.907804593	10.0.0.1	10.0.0.2	UDP	47 55405 → 11000 Len=5
1276	37.907969085	10.0.0.1	10.0.0.2	UDP	47 55405 → 11000 Len=5
1277	37.907655733	10.0.0.1	10.0.0.2	UDP	47 55405 → 11000 Len=5
1440	37.907806036	10.0.0.1	10.0.0.2	UDP	47 55405 → 11000 Len=5
1508	37.907363241	10.0.0.1	10.0.0.2	UDP	47 55405 → 11000 Len=5
31355	52.194527608	10.0.0.2	10.0.0.3	LLC	47 S, func=RNR, N(R)=50; DSAP 0x66 Individual, SSAP 0x60 Response
31357	52.194042196	10.0.0.1	10.0.0.2	UDP	47 55405 → 11000 Len=5
31358	52.194181173	10.0.0.2	10.0.0.3	LLC	47 S, func=RNR, N(R)=50; DSAP 0x66 Individual, SSAP 0x60 Response
31360	52.193597372	10.0.0.1	10.0.0.2	UDP	47 55405 → 11000 Len=5
31459	68.322053462	10.0.0.1	10.0.0.5	HTTP	222 GET /Videos/fauda.mp4 HTTP/1.1

Figure 26: Automatic switch from the first video server to the second one

5.6. Simulation discussion

Although this simulation is successful and illustrates the aspects of dynamicity and autonomy, it mostly shows the feasibility of the proposed architecture. Several other scenarios may be implemented to show the behavior of the solution when several network slices are running. For instance, we can perform some statistics to observe when some slices are affected by the degradation and other are not. Each service or slice can have its own QoS and a link degradation may not be significant for some specific service or slice. Automatic server switching at the network level is not covered today in the current telecom networks. YouTube or Netflix cover such functionality but only at the application layer.

5.7. Conclusion and future research extension

With this simulation, we succeeded in showing the feasibility of the proposed architecture with its enhanced dynamicity (service modification during session), and autonomy (automatic modified service without human intervention). However, several other aspects of this architecture may be studied. What is the behavior of our solution when several network slices are running without the same SLA and QoS? This will require to improve the virtual network service and slice layer. We can also show the ability of this architecture to support E2E and more local constraints simultaneously with the help of this layer. Some scenarios as described in section 4.4, with the dynamic addition/modification of VNFs during session may be also very interesting to demonstrate the interest of our architecture. In our simulation, a maintained and updated application table linked to the application orchestrator will save considerable time in the switch to the new application. However, considering we work in an open sources environment that is very time consuming and is often not well documented or updated with the new OS versions, we have to limit our simulation. Open sources such as OpenStack or OSM also require the use of heavy APIs. Following this work, we realize the importance of the orchestration and its ability to guaranty the autonomy of the network management if an efficient decision-making process is implemented in this part.

Chapter 6. Decision making process

6.1. Introduction: decision-making in case of contention

With the evolution toward the fifth generation of mobile technologies (5G) and the Internet of Things (IoT), future networks will be required to support multi-access mobile technologies, as well as an extremely large range of real-time services with increasing throughput demand. To face such requirements, the network must reach complete autonomy, i.e., become a zero-touch network. Within this context, orchestrators are defined to program autonomic behaviors and support the dynamic changes coming from applications or the network. Previously mentioned surveys on network management requirements have focused on network orchestration [GS1], [26]. To continuously answer dynamic and on-demand application and service requests autonomously, such an orchestration will be required to coordinate the different network entities and constantly make the appropriate decisions. With the growing complexity of telecom networks, human intervention will slow down and disturb the ability of the network to continuously satisfy users service requests. Therefore, the automation of the decision process applied by the orchestrators is a critical issue that will determine the everyday proper operations of the network. For this purpose, a set of predefined rules is used, which are notably decided by the management, planning and marketing departments. However, the application of these diverse sets of rules can lead to additional network delays, conflicts, and even dead ends. The real-time aspects of 5G networks (in which an end-to-end delay of less than 5 ms is expected [27] have become more significant than ever before.

How will network management maintain its autonomy? Under these conditions, how can the orchestrator make the most effective decision and select the most pertinent rule to apply? Our paper addresses this specific problem. Because some rules are of interest under certain scenarios and less so under other situations, how can we distinguish between them?

At the level of the network resource orchestrator, we propose a way to select the most relevant rule when several rules are applicable. We therefore introduce a new decision-making process to improve the use of rules and their selection under resource contention scenarios. After adaptation of this process, other scenarios such as fraud detection or security attacks might be covered. In this context, the originality of our contribution is as follows:

1. A mathematical function that considers all of the included upper-level network constraints, such as billing or SLA, to enhance the decision-making process, at the resource orchestrator level (These top-level constraints are generally not considered.).

2. A mathematical function that helps define limits in the use of resource orchestrator rules. It allows to select the applicable rules and choose the best solution taking into account the priorities (weightings of the parameters). Our solution provides a faster and smarter alternative decision in terms of delay, cost, and quality of service (QoS) degradation for service requests.
3. An improved autonomy in the event of a conflict over the choice of the best rule to apply.

The remainder of this paper is organized as follows: Section 6.2 presents the related work. We analyze different studies related to competitive access to resources, estimation of service level agreement (SLA) for rules weights, and orchestration and its rules. In Section 6.3, we recall our proposed architecture and its functional model in the context of this specific function. In Section 6.4, the problem is described within the architecture detailed in Chapter 3, where the orchestration can improve the decision-making process. In this context, we identify challenging use cases relevant to the resource orchestrator, which require a more efficient decision-making process. In Section 6.5, we explain our methodology and propose a model based on a function used to support the decision-making process. The Section 6.6 describes our choice of parameters, variables, and their value estimation. We also detail a list of existing decision-making processes and rules that can be applied to the described use cases and identify their limits. In Section 6.7, we describe the results of a numerical analysis in which the pertinence of the proposed function is demonstrated. In the discussion section, we discuss the validity of our approach, its dynamicity, and how it may be extended to other scenarios. Finally, we summarize the main details of this analysis and propose areas of extended research, followed by a short list of acronyms used in the document.

6.2. Related work

In this section, we analyze what an orchestrator can bring to network management under certain scenarios. We first survey existing solutions regarding the resource contention problem (see Section 6.2.1). We then identify the gaps that the orchestrator can address, by considering higher-level parameters such as the SLA (see Section 6.2.2). We then analyze articles detailing the contributions of the orchestration and rules to enhance the decision-making process and network autonomy (see Section 6.2.3). Considering the originality of our approach, we extend our research to broader fields unrelated to telecommunications.

6.2.1. Resource contention scenario

To illustrate the interest in using a decision-making function at the level of the network resource orchestrator, we address the problem of resource contention. This case occurs in networks of legacy telephony or quad play operators, particularly during hours of heavy traffic, or during an attack, or natural disaster. This also happened frequently during corona times when the network was particularly busy. Simultaneous concurrent service requests (SRs) attempt to access common and overloaded network resources. In case of resource contention, not all service requests can be accommodated. Although this problem has been largely covered in the literature and can be minimized by applying different methods, it cannot be completely removed from an operator network. This problem appears at different network levels. Ali et al. [28], reviewed a wide range of

proposals for LTE MAC layer congestion control in the case of massive machine-type communications for smart city applications. A novel collision-resolution-based random access (RA) model has been proposed. In reference [29], a continuous optimization method assists with dynamic service admission control in a multi-tier virtual service environment, consequently decreasing the SR rejection. In reference [30], an optimization method was proposed for high-definition multimedia applications in 5G networks to meet the throughput and delay constraints. This article also describes a contention access period supported in 5G wireless technologies: IEEE 802.15.3c and IEEE 802.11ad. This method is a type of scheduled access that provides a dedicated time slot for communication, thus supporting a guaranteed QoS for an application. This guarantee is not possible when all service requests compete for bandwidth using only random access. With the Ethernet protocol, using the Aloha mechanism [31], the packets of a host are sent again after a random period to avoid a new collision. In a user-centric ultra-dense network [32], contention problems appear at the access point (AP) level. For the most part, these methods explore the network context and propose control-plane level solutions. These methods can contribute significantly to limiting the network conflicting resources problem but cannot solve it completely. Although they have a bottom-up approach by applying network and technology parameters, they ignore high-level settings such as SLA. By integrating this type of parameter into the problem, will better results be obtainable?

6.2.2. SLA monitoring

In the context of fifth-generation networks, the SLA parameters allow us to consider the information of the upper layers. These parameters can also support the dynamicity of the environment, such as service changes during a session, at the resource orchestrator to enhance the decision-making process. In this domain, Kosinski et al. [33] are interested in an SLA monitoring and management framework for telecommunication services. They split this procedure into "near real-time" (dynamic) and "long-term" phases. With the help of alarms, performance monitoring (PM), and penalty calculation formulas implemented according to the requirements of the service provider (SP), they calculate the penalties for the SLA violations using the predefined report templates. In addition, Casola et al. [34] integrated security parameters and proposed an automation of SLA evaluation and comparison. Policies are used for both a service behavior and SLA descriptions. A metric matrix function was used to evaluate and compare the policies for the service components. These studies can be extremely useful for estimating the difference between a signed SLA and the real proposed SLA. In addition, our proposition is at the network orchestrator level and allows the processing of relevant information related to the upper level (in our chosen architecture, the orchestration function is distributed per layer) and decides the best rule to be applied according to the application context.

6.2.3. Decision-making and choice of rules at the orchestrator

In real-time management, how can we go beyond the reporting of notifications? To remain dynamic and address the contention problem, our proposal is based on a decision-making function at the resource orchestrator that can support the choice of the most appropriate rule to apply and positively impact the network using the feedback of the network and SLA parameters. In this research domain, telecommunications articles are mostly focused on defining orchestration models

with their rules. In [35], the authors first proposed a platform called Notoriety, located at the southbound interface of the software-defined network (SDN) controller. Its aim is to deliver optimal and filtered information to the SDN controllers, orchestrators, and upper applications, allowing less volume of the receiving data and thus enhancing real-time answers. The authors used a bottom-up approach and did not integrate the parameters of the upper layers, enabling a dynamic contextualization of the solution. In addition, Falelakis et al. [36] dealt with an orchestrator of a video conference system with many video streams, highlighting the need for an intelligent and automatic selection of the most adequate camera views to be displayed on each screen. The defined rules for this purpose, are mostly applied according to the time considerations and the approach is organizational; however, there are no real cases of concurrent or conflicting rules. Moreover, Sonmez et al. [37] proposed a fuzzy logic system to choose the most favorable edge computer for end systems. It can be located locally, remotely, or at the cloud level. This is applied in two stages considering parameters such as the WAN bandwidth, length of the incoming task, and VM utilization, or the delay sensitivity of the computing edge resources. Some rules were applied to efficiently choose the correct location for the application server. These rules integrate the dynamicity of the parameters and the network context. However, we do not question the relevance and efficiency of the rules to be applied in any of these studies. Although several articles have dealt with the need for orchestration and rules within the telecommunication area, to the best of our knowledge, we did not find anything related to the selection of the most relevant applicable rule. Outside of the telecom field, Lammers et al. [38] explored which institutional conditions enable or disable the decision-making processes in the introduction of smart energy systems in four Dutch cities. "The rules of the games" were analyzed, and the need for an adequate orchestration of such rules was shown. This article dealt with the fact that there are numerous rules and stakeholders for decision making and smart energy system introduction, whereas the rules of the game, i.e., who does what and when, are seriously missing. In [39], the Electre method makes it possible to create and choose the best association rules and limit their number to improve the decision making in application areas such as business or scientific studies. One of the ideas is to weigh the rules to select the most relevant ones. Reference [40] proposes an objective function to optimally allocate speed cameras on poles according to road accident data and geographical constraints when fewer cameras than poles are available in the studied country. Parameters such as the number of poles, cameras, movements, and periods are also weighted. This model although in another area, helps us to define our function. However, neither of these approaches completely solves our problem where we seek the best rules to apply in an orchestrator where these rules are already defined. We also want to integrate top-down parameters into our decision-making process. This results in the use of a function that makes it possible to weigh the rules, while using evolving parameters to adapt to the network context and the dynamicity of the services.

6.3. Architectural used for our proposal

We recall here the context of our proposal, particularly the choice of our architecture (see Chapter 3). With this architecture, the intelligence is not located within the network but at the control plane level, particularly in the orchestrator. We also describe the precise functional

approach of this architecture to identify the role of every module and where, within the architecture and its different modules our proposal can be applied.

6.3.1. Reminder of the main elements of the architecture

Two main elements are introduced in the selected architecture (see section 3.3, Figure 6):

- The orchestration is distributed in five layers of network management, where each orchestrator takes care of the corresponding layer, namely, the user, application, network slices and services, resource allocation, and technology layers. This distribution on different layers improves the performance and autonomy and highlights the defined and delimited responsibility of each orchestrator.
- A virtualization layer above the SDN controller allows a SR virtual deployment to enhance the dynamicity of the network management. The network slice and service (NSS) orchestrator controls this additional layer. The referenced study highlights the critical role of virtualization, network slicing, and orchestration in the service provisioning. This virtualization layer offers an answer to the problematic aspect of the end-to-end (E2E) constraints at the application layer simultaneously with more local QoS constraints at the resource layer when provisioning a service. This network virtualization layer with its orchestrator allows the creation of a virtual service with every component without being limited by the physical resources. This allows on-demand component dynamicity without a loss of services.

As detailed in section 3.3.3 the functional model of this architecture is as follows: The user orchestrator qualifies the end-user or SP service request. The application and service orchestrator identifies the end points of the SR and associates the service profile and service type. A weight is also calculated according to the importance of the service. The NSS orchestrator creates a virtual deployment of the complete service with the help of the NSS virtualization layer. This layer is updated with the resource manager and running service databases. In this layer, although the service deployment is virtual, it is complete with its own endpoints, VNFs, virtual path, and upper-layer E2E general constraints, realized at a high level of abstraction, allowing the deployment of a virtual SR. This separation with the physical network resources allows an unloading of the physical network.

6.3.2. Orchestrator of network resources

After going through the three first orchestrators, for the fourth step we define our decision support function. A virtually deployed SR is sent to the general SDN controller (see Figure 27, steps 4 and 5) for a physical allocation. This SDN controller requests the allocation of the virtual resources along with their QoS constraints from the domain or technological controllers. These more specific controllers use the network resource virtualization layer, which calculates the complete path with the VNFs and links. If all resources are available, VNFs are allocated on virtual machines, and virtual network resources are allocated on the network resource virtualization layer. The SR is then placed and activated in the physical network. Once created, the

different orchestrators and databases are updated, and the new service is monitored based on statistics and the PM to provide real-time feedback to the different entities (see Figure 27, step 8).

Under resource contention, what will happen if simultaneous and similar SRs of equivalent weights are sent to the general SDN controller? In this case, the first three orchestrators have already qualified these SRs without filtering because the related users are authorized, and their SRs respect their SLAs. These SRs may be virtually deployed as well, considering the high level of abstraction that exists at this layer. Thus, when we want to allocate resources, the network lacks such free resources in the context of an overbooked entity.

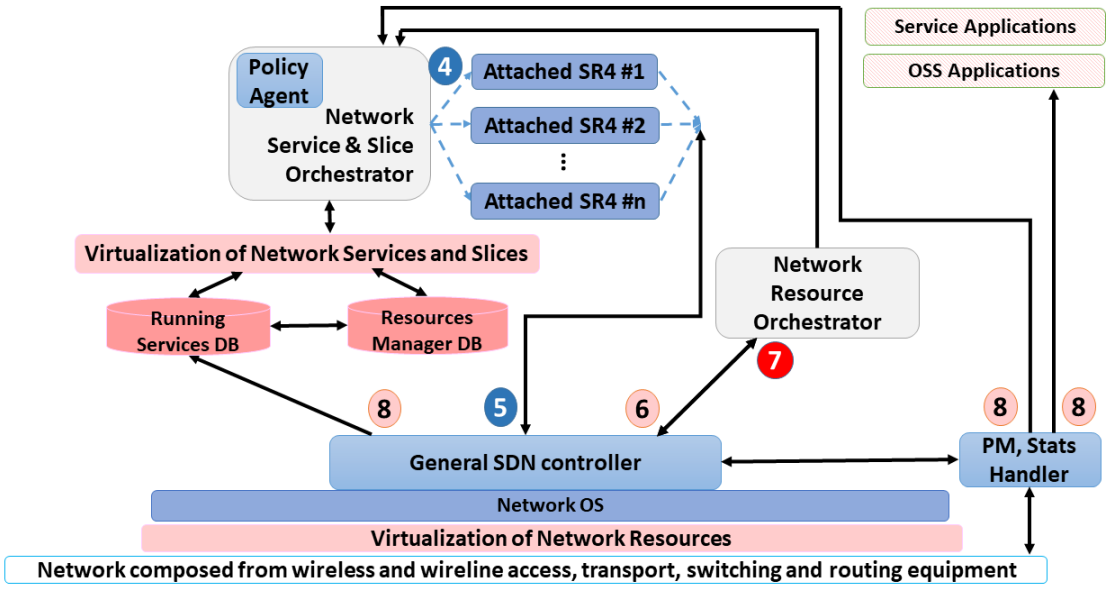


Figure 27: Multiple SRs in overbooked network

Thus, a conflict appears at the resource allocation level. The general SDN controller applies the path computation engine along with its constraints. After path calculation, only a portion of the SRs can be physically deployed. A random decision on *K* SRs among *N* can be applied, and the others are rejected (see Figure 27, step 7). This relatively simple decision-making process can appear to be the most effective. *K* end-users are satisfied, and the network neutrality is respected when considering the random nature of the decision. The different databases are then updated. The PM and statistics handler collects key performance indicators relevant to the performances and quality of the network or service, thereby enabling network changes, optimization, or self-healing. This provides positive or negative feedback to the resource orchestrator. Such indicators allow permanent service monitoring to inform the network and refine the policy of the resource orchestrator if necessary. A second option over a random decision is to turn to the resource orchestrator itself (see Figure 27, step 6). This consists of using rules within this orchestrator, related to the network layer, to try to enhance the number of allocated services. If an applied rule allows more services to be allocated with a good performance, this rule becomes preferable to a

random decision. In addition, if the feedback is positive, the rule continues to be enforced. Positive feedback is a prevalent process in nature, e.g., in human physiology including blood clotting, lactation, or contractions during childbirth [41]. By contrast, the application of rules may generate greater delay or degrade the services, resulting in more unsatisfied customers. This may also result in a more complex orchestrator with more rules, making maintenance more difficult. Similar cases such as the use of concurrent network slice requests for two different SPs can be considered for the third orchestrator.

6.4. Raised problem

This section formulates the problem and the context of the proposed solution. We situate the use case at the level of the resource allocation and the network resource orchestrator. The general SDN controller (see 3.3, Figure 6: Proposed architecture) applies a path computation algorithm, a method to find the shortest path between two points (Distance vector and link state routing [31] are among the most popular approaches used in routing.). The resource orchestrator is then required when the execution of the path computation algorithms is insufficient. This can occur in the case of missing resources or a network failure. The SDN controller ensures the network autonomy permanently during the daily operation. In case of problem, the network resource orchestrator with the help of an information database, can make more accurate decisions based on a set of rules, and thus maintain network autonomy. When N simultaneous SRs must be allocated and the network lacks resources, the general SDN controller is unable to allocate N services and will therefore ask the resource orchestrator for help (see previous section, Figure 27, step 6). There, rules are applicable to support the general SDN controller and increase the number of allowable services. The orchestrator can be useful in several other cases like lack of Virtual Network Function (VNF) in the right place or a need to move a VNF to another virtual machine, but as part of an article we need to limit our subject. Figure 28 details this scenario of an overbooked entity. Here, J rules exist from 1 to J, where j = 1 is for the random rule. When K_1 SRs can be provisioned with $rule_1$, the case is forwarded to the resource orchestrator. The first option is to choose K_1 SRs among N and complete their provisioning if no rule can improve the situation. The second option occurs when an applicable rule j enhances the value K_1: The new number of possible deployed services is now K_j, with $K_j > K_1$. In this case, this rule is applied in the network and the paths in the general SDN controller are re-calculated. If the rule effectively enhances the number of SRs allocable to K_j, the orchestrator is updated with a new number of possible deployed services K_j and the SDN controller allocates these SRs. Updating the orchestrator is important for calculating the network feedback of the rule.

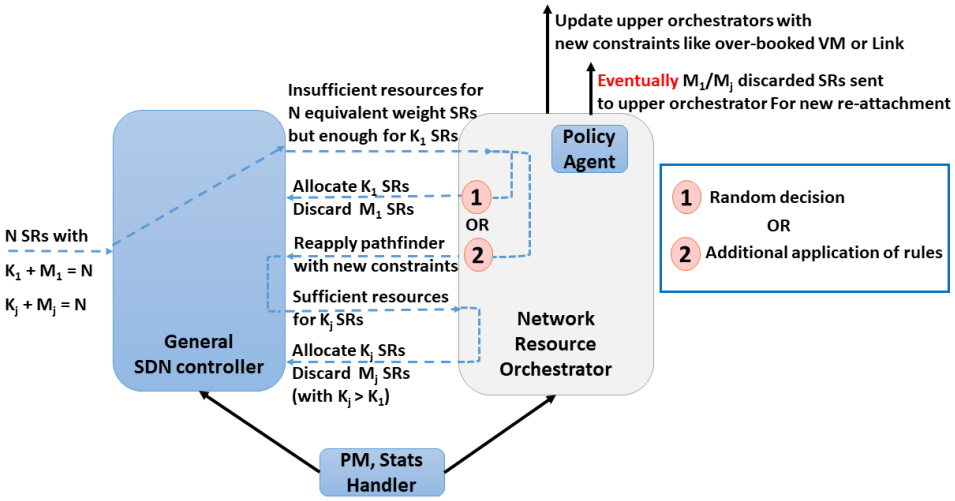


Figure 28: Description of the decision process at the resource orchestrator

In Figure 28, we also consider the possibility of enhancing the number of allocated SRs by re-sending M_1/M_j discarded services to the upper orchestrators for re-qualification and re-virtual deployment with the updated constraint (overbooked entity). This may lead to the selection of another application server, and such a decision will not always be possible and will depend on the operator's policy or technical constraints. Moreover, to avoid an infinite loop under the same tentativeness of the SR allocations, a decreasing flag can be defined in the general SDN controller. Therefore, can every applicable rule improve the number of allowable services? How can the orchestrator make the best decision in a rational and autonomous way? In the next section, we propose a mathematical function supporting the decision-making process.

6.5. Methodology and proposal

6.5.1. Methodology

Following the issue raised in the previous section, when we want to select the most relevant rule to apply for a given scenario, we must first know which rules are applicable. They can be conditioned using an IF command. For each scenario, a set of applicable rules exists. It is therefore necessary to collect all necessary network data to select the best rule. In this case, the larger the knowledge base, the more relevant the decision will be. A mathematical function using these data, can estimate these rules and associate a weight with each of them. Finally, the rule with the highest value is the rule to apply. Thus, to solve our problem, a mathematical function to estimate each rule seems to be the most suitable option for making the best possible decision.

6.5.2. Mathematical function description

The aim of this section is to determine an objective function that will be able to decide whether it is worth using a specific rule instead of a random rule only, and if so, which rule in the context of our defined scenario should be applied. In Section 6.6.3, we detailed a short list of possible applicable rules along with their advantages and limitations. The proposed solution to the

problem described in Section 6.4 considers every network and operator constraint and should allow the most appropriate decision to be made. We therefore define a rational way to quantify the different options and parameters to make the best decision according to the operator's criteria. We first define a general objective function with the parameters to be optimized. These parameters integrate the main variables that can influence a rule to be applied or not in the context of a contention scenario: the number of satisfied/dissatisfied users, the generated additional income or cost, the dynamic feedback of the network regarding every rule, the signed SLA with parameters such as QoS, delay, attributed bandwidth. Considering these parameters, we then propose different constraints and a method to calculate them according to the different types of operators. Considering the objective function in general, the calculations of its parameters are variable and can change considerably according to the operators, regulations, or environmental constraints. The proposed formulas for each parameter can therefore be modified. As an objective function, we propose a function called the total associated value ***TAV (j)***, the principle of which is calculating the value associated with each applicable rule. The rule that presents the highest value is the selected rule. This function is based on other studies using numerical analyses [40]. We made some preliminary assumptions:

- Although several rules can be applied simultaneously, doing so results in greater complexity. Some rules may be contradictory, and some inter-dependencies may exist. Therefore, when executing the mathematical function, only one applicable rule will be selected since one rule can include multiple actions.
- j is the identifier of the rule with $j > 1$.
- $j = 1$ corresponds to the random decision by default.
- $K_j > K_1$, where K_1 and K_j are integers and K_j is the estimated value of the number of allowable SRs after the application of $rule_j$.
- The objective function is the sum of the sub-functions, where each of them is a ratio (unitless), to avoid a problem of consistency in the function.

The ***TAV (j)*** is detailed below:

$$\boldsymbol{TAV(j) = SAR(j) \cdot \omega 1 + IIM(j) \cdot \omega 2 + SVR(j) \cdot \omega 3 - COST(j) \cdot \omega 4 - SLA(j) \cdot \omega 5} \quad (1)$$

where

- w_i is the weight associated to each sub-function. its value depends on the operator policy (see Section 6.6.3);
- *SAR(j)* represents the proportion of satisfied requests;
- *IIM(j)* represents the income improvement ratio by using $rule_j$;

- *SVR(j)* represents the statistical value of $rule_j$ considering the feedback of the network when applying this rule;
- *COST(j)* represents the additional cost ratio when applying $rule_j$;
- *SLA(j)* is a ratio of the estimated SLA when $rule_j$ is applied with the client's signed SLA.

The proportion of satisfied requests is calculated as follows:

$$SAR(j) = \frac{K_j}{N} \; with \; j \geq 1 \tag{2}$$

where K_j is the estimated number of allowable services after the application of $rule_j$.

The income improvement ratio is calculated as follows:

$$IIM(j) = \begin{cases} 0, & if \; j = 1 \\ \left(\frac{K_j - K_1}{K_1}\right), & if \; j > 1 \end{cases} \tag{3}$$

where $\frac{(K_j - K_1)}{K_1}$ is the relative improvement divided by the initial number of allowable services. This improvement does not consider the real number of services. For instance, when $(K_1, K_j, N) = (6, 8, 10)$ or $(60, 80, 100)$, the relative improvement has the same value despite the different number of unallocated services. The relative improvement is nevertheless applicable to every rule.

The statistical value of rule j is calculated as follows:

$$SVR(j) = \begin{cases} 0, & if \; j = 1 \\ \frac{SuccessR(j) - FailureR(j)}{Total(j)}, & if \; j > 1 \end{cases} \tag{4}$$

where *SVR(j)* is a weight associated with the rule based on the success rate (`SuccessR`) of its application. The initial value is `0` before using the PM and statistics feedback. After applying one rule, if the network performance improves, (`SuccessR(j) := SuccessR(j) + 1`), else (deteriorates) (`FailureR(j) := FailureR(j) + 1`). This is based on the feedback of the network and is comparable to the approach used in Q-learning (see section 6.6.3). This approach is simple but can be made more complex by considering the relative importance of the improvement brought about by the rule. This offers the function a type of intelligence, i.e., a constant adaptability of the rules to the network changes.

Note: $SuccessR(j) + FailureR(j) \leq totalR(j)$ and thus, $(-1) \leq SVR(j) \leq 1$

The additional cost ratio when applying $rule_j$ is calculated as follows:

$$COST(j) = \frac{(N - K_j)}{N} + C_j \qquad if\ j \geq 1 \tag{5}$$

where $\frac{(N - K_j)}{N}$ is the relative loss of income from the unallocated services. In addition, $C_j = 0$ when $j = 1$. If $j \geq 1$, C_j can be calculated as a ratio of the required added resources to get more allocated services divided by the resources of the corresponding operator. We propose an estimation of C_j in Table 3 according to the added resources: a VM, VNF, or link, or by going through the network of another operator. If some low priority services must be removed, the cost of the unsatisfied customers must be estimated. This cost can also consider the environmental requirements such as the additional cost of the energy consumption of the added resources. Some countries limit their CO_2 emissions, and when a given threshold per company is exceeded, an extra tax must be paid.

Note: In reference [30], the authors propose a revenue function based on the difference between the revenue and penalty. However, we found it easier to separate the cost from the revenue improvement.

The SLA is calculated in the following manner:

$$SLA(j) = \frac{(SLA - SLA_j)}{SLA}\ , \qquad if\ j \geq 1 \tag{6}$$

where SLA_j is the estimated SLA value for $rule_j$. In addition, SLA represents the signed agreement. As described in related work [33, 34], different methods for calculating the SLA and comparing it to the signed SLA between the customer and the operator, exist. In our function (6), the SLA calculated after applying the rule is compared to the signed SLA (absolute), considering that the random decision SLA (relatively, $j = 1$) is part of the function. Services such as gaming, browsing, or video streaming require a minimum bandwidth to run correctly. To allocate more services, the operator can decide to affect this minimum value despite the customer deserving more bandwidth regarding the signed SLA. Here, SLA_j is a function of security, the real QoS, the effective rate, and the introduced latency per SR. At the application level, SLA parameters such as the availability, maximum time for restoration, or the violation time can be considered. To calculate SLA_j. As an example, we propose Function (7), which illustrates a calculation of latency $LAT(j)$ introduced by $rule_j$. We do not use a complete evaluation of the SLA. Within the context of this study, we estimate that all allocated services respect the SLA.

$$LAT\ (j) = \begin{cases} 0\ , & if\ j = 1 \\ \left(\frac{T_j}{T_{max}}\right), & if\ j > 1 \end{cases} \quad (7)$$

where T_{max} is the maximum delay, at which the use of the rule is no longer justified, and T_j is the evaluated delay introduced by each rule. This delay is composed of several parameters, as detailed in section 6.6.1., we can fix T_{max} as the maximum time that an end-user is willing to wait to receive its services. For example, when making a call, a user will not wait more than 10 s to hear a ringtone. Setting T_{max} has a part of arbitrariness because one second more or less can validate or invalidate the use of a rule. Although this can be part of the signed SLA, it depends considerably on the type of service. The factor $\frac{T_j}{T_{max}}$ decreases the interest of the rule when the delay augments. Thus, if $T_j \geq T_{max}$, the rule is withdrawn.

In the next sections, we describe the choice of the parameters, variables, and rules, and present the numerical analysis along with the results.

6.6. Parameters, variables, and rules for numerical analysis

As in the previously described functions, several parameters depend on the operator, regulation, or economic environment. In this section, we set the values of the parameters ω_i related to the operator type. The variables SVR(j), T_j, and C_j depend on rule j (C_j also depends on the operator type.). We detail the rules related to the allocation resource problem and explain their limitations to better emphasize the need for a rule decision function. Thus, in our numerical analysis within the selected scenario, we essentially apply different rules (SVR(j), T_j, C_j) and variables such as K_j , N, or T_{max} according to the service type. We also manipulate the ω_i parameters by modifying the operator type.

6.6.1. Operator parameter settings

To fix the ω_i parameters, we have defined four operator profiles: internet service provider (ISP), mobile virtual network operator (MVNO), competitive local exchange carrier (CLEC), and incumbent local exchange carrier (ILEC). This covers most of the operator types and can help operators to choose their own values for ω_i. First, $\sum_{i=1}^{5} w_i = 100$ makes it possible to maintain a relative influence of each of the sub-functions. Parameter ω_1, related to the number of satisfied SRs, is important to all types of operators. However, a small MVNO or an ISP with little or no infrastructure, and under severe cost pressure, may not be able to invest in infrastructure, at least when first offering their service. Thus, ω_1 will be lower for MVNO and ISP. For an ILEC or a CLEC, which provide services to enterprises and different ISPs, the number of satisfied SRs is more significant. ω_2 and ω_4 respectively weights of the subfunctions IIM and COST, are directly related to money flows and will therefore obtain similar values. CLEC and ILEC have many physical resources and the IIM and COST may have a limited impact on these types of operators.

Thus, ω_2 and ω_4 will be lower for these operators and even more for an ILEC who usually has a more important infrastructure than a CLEC. However, cost remains the major parameter in most carrier networks. In addition, ω_3 is associated with the positive or negative feedback of the rule and has a value of 10 for each operator because the importance of the rules does not depend on their type. ω_5 is related to the SLA. For an ILEC and a CLEC often working with enterprises, ω_5 is critical and will thus receive a high value. By contrast, for a low cost MVNO, the respect given to the SLA will be less significant, at least at the market launch of the service. In Table 2, we list some practical values of the ω parameters based on the previous principles described:

Operator type	w_1	w_2	w_3	W_4	W_5
ILEC	30	15	10	15	30
CLEC	20	20	10	30	20
MVNO	20	20	10	40	10
ISP	20	20	10	40	10

Table 2: Parameter values according to operator type

Cj depends on the type of operator. It will be more important to an ISP than an ILEC. An ILEC usually benefits from a significant infrastructure and several advantageous agreements with different operators. Thus, the additional network resources required for specific traffic spikes should be less significant and easier to achieve. But *Cj* depends also on the applied rule. In Table 3, we present a simple estimation of *Cj* based on calculation of cost ratios. It results from the use of more resources from the operator's network or from other operator networks divided by the available resources of the operator type in the network. The network sizing of each operator type comes from live network. However, we cannot disclose our sources. When adding a resource, we consider its cost and its deployment. For a VM, the cost is double because the server is often protected with another one. The cost conditions when using the network resources of another operator can differ according to the signed agreement. If they differ from one operator to another, a new column can be added for each operator. In our case, we simplify this by considering that equivalent resources from another operator will cost twice as much as the resources added by the operator to its own network. In Table 3, according to the required added resources, ratios are filled according to the operator type network size and other constraints. We chose resource units that have the same price order to have comparable ratios. We also consider the environment tax as a ratio of the operator taxes. This is very variable according to operator, country, and regulation. Thus, *Cj* is the sum of every additional cost ratio generated when using *rulej*. the ILEC network sizing does not appear in the table. Its network is usually too large and too complex for us to evaluate.

Additional element	Cost ratio from		Network sizing		
	operator	else	MVNO	ISP[5]	CLEC[6]
Network node[1]	2	4	20	20	100
Transit link[2]	2	4	20	20	200
Virtual machine[3]	4	8	150	250	100
Memory unit[4]	2	4	50	150	60
Environmental tax	5	NR[7]	100	100	250

[1] A switch router of 24 ports is considered as a basic unit.
[2] We consider a 100 Mbps transit link extension, which is negotiated with an operator.
[3] Each virtual machine uses 8 cores and 10 GB RAM to run.
[4] Each memory unit consists of 256 GB SSD.
[5] We described here a hosting and mailbox ISP.
[6] Here is detailed a regional CLEC. The transit link may belong to him.
[7] Not Relevant.

Table 3: Calculation of C_j, sum of cost ratios

6.6.2. Estimating rule variables

In the previous section, we proposed a method to calculate C_j. In this section, we focus on the second variable T_j, related to the delay generated by rule j in $LAT(j)$. For this purpose, we compare the time computing complexity of using rules owing to the real-time constraints. The space complexity changes from one operator to another and is ignored. We then make an estimation of variable T_j. We call $TD(j)$ the **time duration** required to allocate K_j SRs when using $rule_j$. This is expressed as the sum of the following different delays:

$$TD(j) = D_{PM} + D_{df} + D_{rulej} + D_{Kjpc} + D_{KjAl} + D_{rKj} \quad (8)$$

where

- D_{PM} is the PM monitoring delay used to understand the reason why only K_1 SRs can be allocated,
- D_{df} is the delay related to the execution of the decision function in order to estimate the best $rule_j$ to apply,
- D_{rulej} is the delay related to the execution of the orchestrator's $rule_j$,
- D_{Kjpc} is the delay of the path computation for K_j SRs (only applicable if $rule_j$ improves the number of allocated services),
- D_{KjAl} is the delay in trying to allocate K_j SRs using the path computation algorithm (instead of K_1), and
- D_{rKj} is the delay in randomly choosing K_j SRs among N.

If we calculate *TD(1)*, i.e., a random rule, the sum of the different delays is as follows:

$$TD(1) = D_{PM} + D_{df} + D_{K1Al} + D_{rK1}. \quad (9)$$

The main difference lies in the fact that a rule j is not executed because the decision function does not find any favorable rules in this case. Thus, the path computation does not have to be run again for K_j SRs. Thereby, we express T_j as the difference between $TD(j)$ and $TD(1)$ with $T_1 = 0$. Here, several parameters of T_j are not of consequence. In this case, D_{PM} and D_{df} disappear from the function below. The delay in choosing K_j SRs among N when using the Fisher Yates shuffle algorithm, for instance, has a time complexity of $O(K_j)$. Thus, in most T_j estimations, $(D_{rkj} - D_{rk1})$ can be neglected. The difference in delay between allocating K_j or K_1 SRs, is a part of the network life. Allocating more SRs is the aim of the operator despite it being more time consuming, which should not influence the decision. Therefore, we can express T_j as follows:

$$T_j = TD(j) - TD(1) = D_{rulej} + D_{Kjpc} + \left[\left(D_{KjAl} + D_{rKj}\right) - \left(D_{K1Al} + D_{rK1}\right)\right] \sim \boldsymbol{D_{rulej} + D_{Kjpc}} \quad (10)$$

where D_{rulej} depends on the rule itself and will be determined for each rule in the numerical example. In addition, D_{Kjpc} depends on the chosen routing algorithm. For instance, algorithms such as A* and Dijkstra's algorithm achieve lower time complexities of $O(|E|log(|V|))$, where V (vertices) is the number of nodes, and E (edges) is the number of links between nodes, which must be applied K_j times. Therefore, D_{Kjpc} has a time complexity of K_j*$O(|E|log(|V|))$ depending on the network size. However, translating it into the duration is difficult and depends on the allocated CPU and memory resources as well as the type of application. Therefore, we use an estimation based on different articles, i.e., a constant value of 20 ms for a path calculation [42, 43]. Even in huge networks (with approximately 100,000 V) [44], the value is less than 1 s. Shorter times should be obtained with micro-services [45] that decorrelate the treatment from the logic of the sequences.

Therefore, in our use case, an additional rule execution of the resource orchestrator will always be more time consuming than a random rule because of D_{rulej} and the additional path computations of K_j SRs. This appears to be of interest if the time required by the end-user to obtain service, remains within "tolerable" limits i.e., a waiting time that the customer can bear. The QoS of the K_j deployed services must remain "acceptable." This information is provided mainly by the statistics and PM handler when the network feedback is positive.

6.6.3. Rules and their limitations

A decision-making process allows us to automate the best choice between a predefined policy applied at the resource orchestrator and the default option of randomly rejecting some services given certain resource constraints. For this purpose, we focus on rules that are deployed in the policy agent of this orchestrator, relevant to the case of resource contention, and used in our numerical analysis. Other rules can be found in an already published document ("Rationale for

using of a decision support function at the level of the resource orchestrator", at the following url: shorturl.at/vLMQ9). The use of these rules often has certain limitations, which can make them unwarranted. Thus, the following rules detail the need for a decision process:

• **Use of additional resources**: The SP allocates more network resources such as CPU, memory, or bandwidth, to satisfy more SRs, although this generates additional costs.

• **Network slicing** is a strategic part of 5G network architecture [22]. In this context, the network slice is temporarily re-dimensioned: A slice extension or use of another slice can allow the allocation of more services, even according to the time intervals. However, the use of another slice can be a security issue, a link can also be overbooked in the other slice, and the setup of a slice extension can generate additional delays.

• **Energy saving**: Relative rules can activate more resources when the number of allocated SRs is increased. A simple solution as a background application can deactivate certain virtual machines (VMs) or antennas to reduce power consumption. Network virtualization also saves energy. Reference [46] addressed the problem of an energy-efficient orchestration of online service function chaining requests across a multi-domain network without violating the privacy demand of multi-domain networks. This article also proposes the optimization of the orchestration energy consumption of such a process. Owing to its awareness of the entire network, the resource orchestrator can avoid some domains or go through the same VNF multiple times. Such an application can be complex to implement and deploy because of its multi-domain support. Moreover, a similar application is considered to exist within each domain.

• **Artificial-intelligence**-based machine learning is resource consuming.

o **Deep learning** for automatic rule generation: impressive results have already been obtained in autonomous cars, medical diagnoses, and defect detection for new code error prediction with advanced algorithms [13, 47]. In the telecom field, the fault detection and prediction of impending failures [48] and intrusive detection systems [49] have shown encouraging results.

o The **Q-learning** method can improve the rules and decision process when the rules themselves are weighted according to network positive feedback (PM and short- and long-term statistics) [50]. In the case of multiple applicable excluding or conflicting rules, those with a higher positive feedback, will be applied first. This has been applied in heterogeneous 5G networks for the self-configuration/optimization of femtocells. The model should solve both resource allocation and interference coordination problems in the downlink of femtocell networks [51]. However, this case may be more appropriate for technology orchestration. In addition, to achieve full autonomy, management functions must be able to interact easily with managed resources.

Although the above rules can enhance the network performance, dynamicity, and autonomy, their application may also generate more complexity, cost, delay, and conflicts. Thus, continuous monitoring is required before applying them. In the case of positive feedback in a network, such as the allocation of more services or no QoS degradation, the orchestrator continues using these rules. Determining the network feedback remains difficult when considering that

several parameters, such as the number of allocated services, QoS, operator cost, and network availability must be estimated and combined. Its calculation also depends on the operator policy and the regulations of the specific country.

6.7. NUMERICAL ANALYSIS EVALUATION

In our numerical example, after specifying additional restrictions, we conducted numeric analysis in two main directions: putting different rules into competition, choosing a rule, and modifying its variables K_1, K_j, N, T_j, or type of operator. Numerical analysis is performed by applying different values to the parameters and variables of the objective function for each selected rule. every time, we check what the calculated value of the rule is and whether it is worth using it.

6.7.1. Restrictions in the application of the function

The additional limits are as follows:

- If $K_j \leq K_1$, only a random rule is applied;
- $K_j < N$ with $K_j \# N$ *because there are always new-coming SRs.*

Note 1: We consider that when K_1 SRs among N are allowable, the general SDN controller will wait for feedback from the resource orchestrator and allocate K_j SRs globally owing to the use cases we have chosen. In certain cases, however, first allocating K_1 SRs and then allocating $(K_j - K_1)$ SRs only after the application of the rule may be reasonable.

Note 2: A rule that appears unjustified in real time may prove to be based on long-term based. Adding resources may cause T_{max} to exceed. In the longer term, with the help of this resource extension, the case of missing resources can disappear in this part of the network. This can be decided by the strategic and planning departments, based on the feedback of the TAV function and on the long-term PM and statistics at a higher orchestrator or OSS level.

6.7.2. Illustration of the decision function with different rules

In this first analysis, we apply the function to five different rules. Each rule must be evaluated. As described in the Section 6.6.3, if the rule consists of activating additional resources such as a link or node, or if the network slice is resized to meet the new service requirements, it should not exceed the time required for a service provisioning or modification. If some VMs are modified or migrated to another physical machine, the generated delay here can then vary according to the application or VNF. It can be estimated according to the historical statistics of the network or tests realized earlier.

We proceed with two different simulations: one with a CLEC and another with an MVNO. We fix the parameters as $K_1 = 50$, $N = 100$, and $T_{max} = 8\ s$ for a video call service and use the parameters in Table 2 and Table 3. The value of $SVR(j)$ depends on the feedback of the network

and can change during the life of the network. *LAT(j)* is the result of the equation (8) and (11) and depends on the values of K_j and $Drule_j$. The result column is obtained by the equation (1) using the parameters of the previous columns. The results *TAV(j)* appear in the last column of these tables:

j	Rule name	*STA(j)*	D_{rulej}	C_j	K_j	*LAT(j)*	Result
1	Random decision	0.3	0 s	0	60	0	3.00
2	Cost[1]	0.7	0.3 s	0.11	80	4.75	15.62
3	Machine learning[2]	0.6	0.8 s	0.12	75	5.75	9.15
4	Environment[3]	0.3	3 s	-0.25	65	10.75	3.92
5	Slice modification[1]	0.7	0.6 s	0.1	84	5.70	18.30
	Decision of the function:					***Rule5***	**18.30**

[1]Activation of additional resources (see Table 3 for C_j estimation)

[2]Mostly computing delay

[3]Rerouting services to decrease resource use to avoid additional tax to be paid.

Table 4: Function usage to decide the best rule to use with CLEC

j	Rule name	*SRV(j)*	D_{rulej}	C_j	K_j	*LAT(j)*	Result
1	Random decision	0.3	0 s	0	60	0	-1.0
2	Cost	0.7	0.3 s	0.27	80	2.38	8.49
3	Machine learning	0.6	0.8 s	0.08	75	2.88	9.93
4	Environment	0.3	3 s	-0.5	65	5.38	18.29
5	Slice modification	0.7	0.6 s	0.4	84	2.85	6.55
	Decision of the function:					***Rule4***	**18.29**

Table 5: Function usage to decide the best rule to use with MVNO

We can observe in Table 4 and Table 5 that despite using similar values, the results differ between an CLEC and a MVNO. In the case of CLEC, the slice modification rule appears to be more interesting because of the low importance of the additional resources (see parameter ω4 in Table 2). In the case of MVNO, the cost is more significant (a different value of ω4) and therefore the environment rule is selected because the MVNO will pay less taxes by using less resources despite the high rule delay.

6.7.3. Modifying variables and sensitivity analysis

In this simulation, we first modify K_1 and K_j. The $N = 100$ and $T_{max} = 8\ s$ values are fixed. We use the "cost" rule using the values of Table 3. When the operator is an CLEC, we obtain the following graph (see Figure 29): When the line becomes positive, it indicates that it is worth using the "cost" rule. We can observe that the evolution is relatively linear. When K_1 has a higher value,

with a fixed value of *Cj*, a relatively fixed enhancement of $(K_j - K_1)$ is necessary to justify the use of the rule. If C_j is lower, this value also decreases. If K_1 is higher, K_j should be also bigger to justify the use of the rule: $(K_j - K_1) = 3$ when $K_1 = 50$ and $(K_j - K_1) = 6$ when $K_1 = 80$.

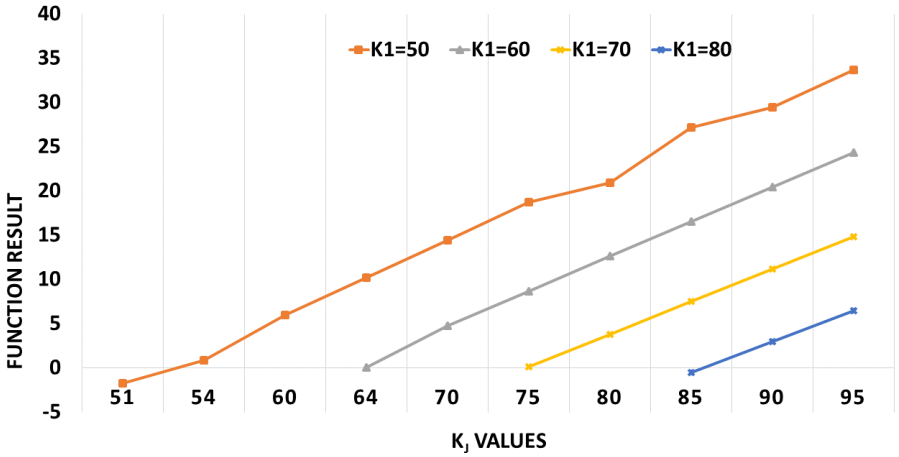


Figure 29: CLEC decision results when modifying K_1 and K_j variables

If we use $K_1 = 500$ and $N = 1000$, we elaborate a more complex function: If we want to maintain a delay of 0.02 s per path computation, we must add some CPU and memory resources. This same delay is applied to more SRs, although the cost increases because we need more resources. We added a cost per VM unit according to Table 3. As we can see in Table 6, we need 256 SRs to be added to justify the use of the cost rule.

K_j value	502	700	750	**756**	800
TAV(j) value	-17.37	-4.05	-2.52	-1.98	1.95
TAV(1) value	-2.00	-2.00	-2.00	-2.00	-2.00
TAV(j) – TAV(1)	-15.37	-2.05	-0.52	0.02	3.95
Decision:	$Rule_1$[1]	$Rule_1$	$Rule_1$	$Rule_j$	$Rule_j$

[1]The random rule usage is preferable

Table 6: Function usage with CLEC when K1 = 500 and N = 1,000

When modifying the delay, particularly with the parameter D_{rulej}, in Table 7, we can see that if the delay overcomes a value of 5 s ($T_{max} = 8$ s) for $K_j = 62$, the usage of the cost rule is already not justified.

K_j value	52	55	62	62	62	62
T(j) value	0.1	0.5	2	4	5	6
TAV(j) value	0.25	1.95	4.5	-0.50	-3.00	-5.50
TAV(1) value	-2.00	-2.00	-2.00	-2.00	-2.00	-2.00
TAV(j) -TAV(1)	2.25	3.95	6.50	1.50	-1.00	-3.50
Decision:	$Rule_j$	$Rule_j$	$Rule_j$	$Rule_j$	$Rule_1$	$Rule_1$

Table 7: Function usage with CLEC and increasing delay T(j) (K_1 = 50 and N = 100)

In the second simulation, we use the same variables but with the parameters of the MVNO. We thus obtain Figure 30 as follows: In this case, a (K_j – K_1) enhancement ranging from approximately 9 to 12, is necessary to justify the usage of the cost rule. As we explained, the MVNO is more sensitive to the cost and has corresponding value parameters.

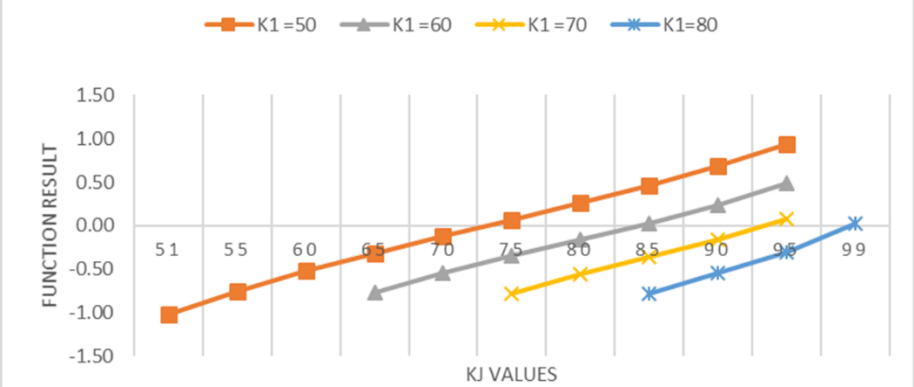


Figure 30: MVNO decision result when modifying K_1 and K_j variables

When trying to use $K_1 = 500$ and $N = 1000$ with the MVNO, the cost of the additional resources becomes so important that the value of *Kj* does not have much influence. In this case, the random rule always obtains a higher value.

The results when modifying three rules, i.e., the randomness, cost, and environment, are shown in Table 8.

Kj value	51	52	54	65	70
Random *TAV(1)*	-2.00	-2.00	-2.00	-2.00	-2.00
Cost *TAV(2)*	-3.70	-2.85	-1.15	8.20	12.45
Environment *TAV(4)*	-3.95	-3.10	-1.40	7.95	12.20
Decision:	*$Rule_1$*	*$Rule_1$*	*$Rule_2$*	*$Rule_2$*	*$Rule_2$*

Table 8: Function usage with CLEC when modifying three rules (Kj = 50 and N = 100)

The environment rule cannot improve K_j as with the cost rule when considering that fewer resources are used and that optimizing the route utilization can generate an important delay. Thus, when K_j = 54, the cost rule becomes more interesting than the random rule, the environment rule fails to become more interesting unless the delay decreases.

6.8. Discussion about the results

Although the approach introduced by this rule decision function involves adding complexity to the orchestrator, this function remains linear and simple and should not add a significant delay to the execution time. This function provides a transverse solution that considers every parameter of influence in the decision making. The results are consistent and convergent. This analysis shows the feasibility of such a function being used to support the decision-making and is added to existing solutions to our contention problem. The TAV(j) function can be easily adapted to best meet the operator's dynamic policy and strategy over time, owing to the flexibility of the parameters values and sub-functions. However, considering the important number of parameters, variables, and possible sub-functions, there are no clear answers to this specific problem. Nevertheless, this is also a guarantee of adaptability for our function and its ability to respond to all types of operators. This function may also be extended to other scenarios such as fraud detection or security attacks. In each case, the determining parameters must be identified along with the corresponding sub-functions. These scenarios can be relevant if several concurrent rules can be applied simultaneously. However, maintaining the sustainability and survivability of autonomous networks remains a mandatory goal. In this context, we have shown that this defined decision function can help the network efficiently choose the most appropriate rule both dynamically and autonomously.

6.9. Conclusion: decision-making function within the orchestrator

When deepening the network management of 5G and IoT networks, the role of the orchestrator appears to be strategic and complex. An orchestrator should not be just an advanced controller. Several decision-making processes and rules will co-exist. It will have to be an intelligent module capable of adaptation, to maintain the network autonomy and make conscious decisions. In this article, we have chosen to base ourselves on a dynamic and distributed architecture, where the orchestration function is distributed among five layers of network management. In this context, we focused on the network resource orchestrator. We illustrated the need for complexity of its decision process with a specific use case where different applied rules can slow down the network or even bring it to a dead end. We have shown that many options and rules need to be defined for this orchestrator and maintained dynamically to guarantee the survivability and sustainability of the network.

State-of-the-art research is still focused on implementing orchestrators and their rules and showing their contribution to network autonomy and efficiency. We introduced a new decision-making function to select the most adapted rule for application in the context of a contention scenario. With the help of a numerical analysis, we have shown the main benefits of its use, which is extremely flexible and can be adapted to all types of operators; it allows an optimization in the choice of applicable rules and thus, improves the network efficiency, which helps keep the network operating, autonomous, dynamic, and adaptable to the changes in the operator, market, network, technologies, and regulations. The originality of this article lies in the introduction of a rule decision function at the orchestrator level and rules related to the business, regulations, or

policies of the operator at the level of the resource orchestrator. However, we only covered a limited number of cases in this study. Because of the large number of parameters and variables, this function requires a significant effort in terms of the parameter settings, while guaranteeing the flexibility. The successful use of this function relies on network feedback, which is extremely CPU and memory resource intensive. However, this is necessary for most 5G network functions. As a future study, we can extend the application of this function to multi-domain support. With the same methodology, this function can also be extended to other scenarios, such as fraud detection. However, to add a relevant sub-function, the relative parameters, rules, and their interactions have to be clearly understood. This programmable function can also be used as a basis for simulating the collaboration between the controller and the orchestrator in an autonomous network, to meet the dynamic and evolving demand of 5G.

Chapter 7. Conclusion and Perspectives

7.1. Overview on Contributions

Considering the exploding number of devices connected to the Internet network, the growing diversity of the services, the network complexity increases continuously with multi-technologies, multi-vendors, multi-abilities network and computing components. In this context, the network requires autonomy, dynamicity, and more intelligence: Autonomy, to avoid human intervention that can be source of error; dynamicity, to make the service request user-centric and on demand, to limit the operator intervention and make the services more cost-effective; more intelligence, to guarantee the autonomy of the network and clever decision-making in the event of conflicts or other specific scenarios. Therefore, our research, in the field of 5G and IoT network management, contributes to these three network attributes, particularly at the level of architecture, organizational and functional models, and orchestration, recognized as the decision-making center of network management.

7.2. Global Contributions

Following a comprehensive study of orchestration and the place of NMS/OSS in the future management of the network, we have identified the gaps and challenges in this area: the demand for open and standard APIs and the migration to the future network management, can only be performed in SDOs and operator NOCs (network operation centers). Therefore, our research has focused on introducing new architecture elements and intelligence in network management. The goal was to address the other gaps identified: user-centric and network-centric dynamicity, orchestration distribution, and network autonomy. These functions help in the deployment of a NaaS (Network as a Service) in the context of the 5G and IoT networks as proposed in the clouds with the SaaS, IaaS, or PaaS. Thus, our three contributions were as follows:

1. To allow the dynamicity and especially the user-centric one, we proposed an enhanced network management architecture and organization model, with the addition of a network virtual layer to create virtual service before its deployment. This created virtual service is a service composition, made up of virtual nodes and links and can be modified dynamically. It can integrate the E2E and more local constraints without the limitation of the more physical resources. This additional layer allows the allocation of other virtual network functions during a service session, by previously creating the modified virtual service before allocating its new/modified resources and switching to the new updated service. We also proposed a distribution of the orchestration in 5 layers, User, Application and service, Network service and slice, Network resource, and Technology, to delimit and

isolate the responsibility of each orchestrator and thus, to ensure the autonomy and efficiency of each layer. This results in an enhanced network efficiency. The objective is to get closer to the zero-touch network.

2. Our second proposal is functional and a continuation of our previous work. It allows to better identify the different functions and their interactions with the help of various use cases and sequence diagrams. This allows us to propose new proactive processes to ensure the E2E QoS of each slice, for instance. Its show the advantages brought by our new architecture. We show by a simulation how our proposal can enhance the user-centric dynamicity during session. We create the need for a dynamic change in a live service and see that without our virtual network layer, the service has to be recreated with a service interruption, whereas this is not necessary if the management of the network benefits of our proposed architecture. To this end, we have deployed an Ubuntu platform with open sources such as Mininet and ONOS and show how by creating the service in two steps, virtually and then physically, it is possible to modify it online ensuring the continuity of the service.

3. Our third proposal was to introduce more Intelligence in the orchestration. For this purpose, we created a mathematical decision-making function to choose the best rule to apply in case of contention at the level of the resource orchestrator. This function considers dozens of parameters and variables such as the types of operators or services, regulations, SLA and QoS, or the ratio of allowable services. For each applicable rule, a score is calculated also considering the added income or cost, and the rule with the higher score is applied to enhance the number of allowable service requests. With the help of a numeric analysis, we show, this function can maintain orchestration autonomy in case of contention and enhance the performances of the network.

7.3. Research Directions for Future Works

As further steps, several directions can be considered. Regarding our architectural and organization proposal, the orchestration is still in its infancy and network slicing can be seen as a mean to really achieve service assurance for different and secured network slices to propose new services in the domain of e-health, industry, or transportation.

Regarding the user-centric and service dynamic composition with container usage, our simulation can be enriched with a more complete platform, new services, and additional use cases. Several questions also arose when working on the simulation: Can we respect 5G required delay by moving VNFs to fulfill on-demand services? How to isolate and secure the different network slices from each other? How to intelligently deal with the huge amount of data coming up from the network? How to ensure effective communication and cooperation between the different orchestrators while maintaining the independence of each layer?

Concerning the introduction of intelligence within the network management, more elaborated function can be created in different orchestrators to include more rules related to security, fraud, or emergency, or more complex sub-functions to calculate the weight associated to each applicable rule. Security and fraud differ seriously from contention, and a mathematic function may not be the best solution to solve these issues. Machine learning can be also an interesting domain to investigate despite its usage has to be justified.

For each one of the considered problems, the road is open for further research. The objective remains to improve the existing network management considering zero-touch network is still a quest for the grail.

Chapter 8. List of Acronyms

Acronym	Elaboration
5G	Fifth generation of mobile networks
ABNO	Application-based Network Operations (IETF)
A-CPI	Applications-Control Plane Interface (ONF)
ALTO	Application-Layer Traffic Optimization (IETF)
AP	Application Plane
API	Application Program Interfaces
BSS	Business Support System
CIM	Common Information Model
CORD	Central Office Re-architected as a Datacenter
COTS	Commercial off-the-shelf in IT world
CP	Control Plane
DP	Data Plane
E2E	End-to-End
ONAP	Open Network Automation Platform
EMS	Element Management System
ENI	Experiential Networked Intelligence from ETSI
ETSI	European Telecommunications Standards Institute
IaaS	Infrastructure as-a-Service
IETF	Internet Engineering Task Force
IoT	Internet of Things
KPI	Key Performance Indicators
LSO	Lifecycle Service orchestration (MEF)
MANO	Management and orchestration
MCC	Management Control Continuum (ONF)
MEF	Metro Ethernet Forum
MP	Management Plane
NaaS	Network as-a-Service
NBI	Northbound Interface
NFV	Network Function virtualization
NM	Network Management
NMS	Network Management System at the NMS layer (not the management of the network that is more global)
NS	Network Service
NSS	network slice and service (orchestrator or virtualization layer)
OAM	Operation Administration, and Maintenance
ODL	OpenDayLight (ONF)
ONOS	Open Network Operating System
ONF	Open Networking Foundation

OS	Operating System
OSS	Operations Support System
PaaS	Platform as-a-Service
PCE	Path Computation Element (IETF)
POC	Proof of Concept
QoS	Quality of Service
SVR	Statistical Value of a Rule
SaaS	Software as-a-Service
SBI	Southbound Interface
SDN	Software Defined Network
SDO	Standard Development Organization such as ETSI, ONF, IEEE, ITU-T, IETF, TMF, or MEF
SLA	Service Level Agreement
SP	Service Provider
SR	Service Request
SW	Software
TAV	Total Allocated Value to a rule
TMF	Tele management Forum
TOSCA	Topology and orchestration Specification for Cloud Applications (OASIS)
VM	Virtual Machine
VNF	Virtual Network Function
XaaS	Everything as-a-Service, i.e., PMaaS for Performance Monitoring and FMaaS for Fault Management
XOS	XaaS operating system
ZOOM	Zero-touch orchestration, Operations and Management (TMF)

Chapter 9. Appendix State of the art: Application orchestration and OSS in 5G networks

In this chapter, we recall the survey that we carried out in the field of network management to better understand the state of the art [GS1] and how to direct our work on the current concerns of research and industry. When we started our research, several SDOs and researchers already have specified and implemented some technologies like SDN or NVF. That's why we decide to concentrate our study in the domain of the orchestration and the role of the NMS, OSS/BSS in the network management of the future for next-generation networks that still was in its early days. This study is included as an appendix because only a part of it is directly related to our proposals, although this step was necessary to position us in this very active field of telecoms.

Several articles and forums deal with architecture and mostly focus on some technologies and aspects of the architecture such as SDN, their southbound interfaces, and the Network Function Virtualization (NFV). However, fewer articles describe the northbound interface and distribution of features between the applications, OSS, BSS, and the orchestration. The CP and the OSS are located at a different layer. If the CP allows introduction of dynamicity in the services, the OSS can go on managing more static services and configurations of the network. The OSS has to integrate flexibility to be able to monitor and bill the new service types and VNFs. So, the CP introduction on the Network Management System has a certain, but limited, influence on the OSS. However, the orchestration at the customer, application, and service levels, will influence the OSS more significantly. This orchestration will have to manage every kind of service and VNF. Thus, we decide to cover the standardization efforts in this domain and the interactions of this orchestration and the OSS. We also show the influence of this orchestrator with the OSS and the applications of this layer and the several challenges present at this level.

Therefore, this chapter is organized as follows: section 9.1 reviews the current architectures and the respective places of the application and service orchestrator and the OSS in the different SDOs. In section 9.2, we make a short review of the main trends in the research with a description of new technologies such as NFV, or orchestration; some European projects demonstrate the feasibility of these technologies; and we also consider the growing demand and need for user and network centric E2E service management (We do not recall the technologies that are already described in Chapter 2.). In section 9.3, we select dynamicity, adaptability, and flexibility as key parameters of our analysis. In this context, we show how this orchestration impacts the OSS and Management Plane over the short and longer term in order to get a more general view of orchestration. We also address the topic of hybrid OSS as a necessary step between legacy

management and future OSS. In the section 9.5, we identify the main features of the application orchestrator and how they will affect the OSS. In the last section, we identify several open challenges of this orchestrator: integration, convergence, synchronization, dynamicity, as well as different options for policy management and for decision-making. Finally, we conclude with some suggested research directions to achieve automation or zero-touch objectives. These suggestions are summarized in the section 2.9.

9.1. Forums and standardization efforts on orchestration

A growing number of forums, industry vendors, operators, and universities are active and collaborate in the SDN and orchestration standardization efforts and implementation. everyone comes with a different approach and proposes differentiated and attractive solutions. Considering the size and rapid evolution of this industry, this survey can hardly be exhaustive, and we present only the most influential forums and research initiatives with regard to orchestration. We first summarize the work accomplished by ONF and then focus on the IETF, ETSI, MEF, and TMF forums. For each one, we present their architecture, their perception of the orchestration and some of their major implementations. We also add a few research considerations related mostly to some new technologies that are not described in chapter 2, some realized European project, and user and network centric service requests.

9.1.1. Open Networking Foundation (ONF)

Open Networking Foundation (ONF) is dedicated to promotion and adoption of SDN through open standard developments supported by main private vendors and telecom operators like Vodafone, TATA, or Telefonica. ONF with OpenFlow™ protocol is first present in the definition of standard and open Interface between the elements and the SDN/NFV layer with specifications of Common Information Model (CIM). By Open and Standard interface, we mean Open for completely documented (including vendor proprietary implementations) and Standard for defined and specified in a forum or institute supported by several major actors in the telecom industry. Even if hardly active at the application layer, we talk about ONF, considering its importance in the industry. Originally developed at Stanford University, OpenFlow, well received in the industry, tries to cover every telecom technology, such as routers and transport with optical or radio links.

For instance, the OpenFlow switch specification is already mature and in its version 1.5.1. [52]. ONF is prolific and several POC have been realized. In the first POC [53], the idea is to prove OpenFlow interface can be specified and implemented for a wireless transport SDN in a multi-vendors environment (e.g., Ceragon, Ericsson, Huawei, NEC, and SIAE) under the umbrella of Telefonica. The SDN platform is based on open network operating system (ONOS) open source. The second POC [54], also in this domain, includes the participation of AT&T and Deutsche Telecom. Some tests are realized on automatic detection and configuration of new microwaves links, and on detection of aberrances. These tests are based on an OpenDayLight (ODL) CP and

NETCONF protocol with an adaptor to the different vendors EMS interfaces. Together with these POCs or others [55, 56, 57], ONF has defined specifications for wireless links, optical transport OpenFlow [58, 7, 59, 60] with a relatively complete information model [61]. For now, in the optical domain, it is possible to interface directly with upper SDN layer with OpenFlow even if the interface between the network elements and the first abstraction level / EMS stays proprietary. However, some native OpenFlow interface has been implemented [62]. Even Specification extension has been defined for MPLS-TP [63]. Concerning its SDN architecture approach [64], ONF foundation is based on the same classical approach with the DP, CP, and AP planes, but OSS runs at every plane with a coordinator module with the Control and Data planes. Mostly focused on OpenFlow interface between DP and CP, ONF is of limited interest at the Application orchestration level. Consequently, Management OSS keeps an important place in the architecture at every management layer: for initial setup, CP policy configuration, performance monitoring, service level agreement (SLA), and security issues. Following more recent evolution of the industry standards (2016, 2017), ONF has extended and updated its SDN architecture view: orchestration place and role is better identified [65, 66]. In these more recent documents, Service management interfaces Northbound with SDN controller at the Application (see Figure 31, red color) layer and southbound with Resources group [67]. The SDN controller acts as a server and satisfies clients' requests by virtualization and orchestration of its related resources. SDN works continuously and dynamically. MP and CP are integrated and convergent and now viewed as a continuum. The aim is to propose each function as a Service (XaaS). For instance, for PM as a Service (PMaaS) or Fault Management as a Service (FMaaS), the collected data are available in a presentation layer for every upper application according to its own needs even if these data come from two different planes. For example, performances of network slice, router, or specific network technology, are available at the same interface, but filtered according to the needs of each one. Thus, we can differentiate between network or resource performances. This concept is referred to as Management Control Continuum (MCC) in ONF. In XaaS, every module is responsible for a specific function perceived by its subscribers as a service. The SDN controller role is to serve continuously client/service request, check state of the requested resources, and after policy application, return answer to the client and thus accept, deny, or modify resources state. This is done with the SDN controller feedback intelligence in a recursive way. In this vision, a common virtualization and orchestration allows relationship between the different CP entities and the different resources groups. The idea of a logically centralized control multi-domain / multi-technology SDN controller is applicable to this architecture and also to other forums.

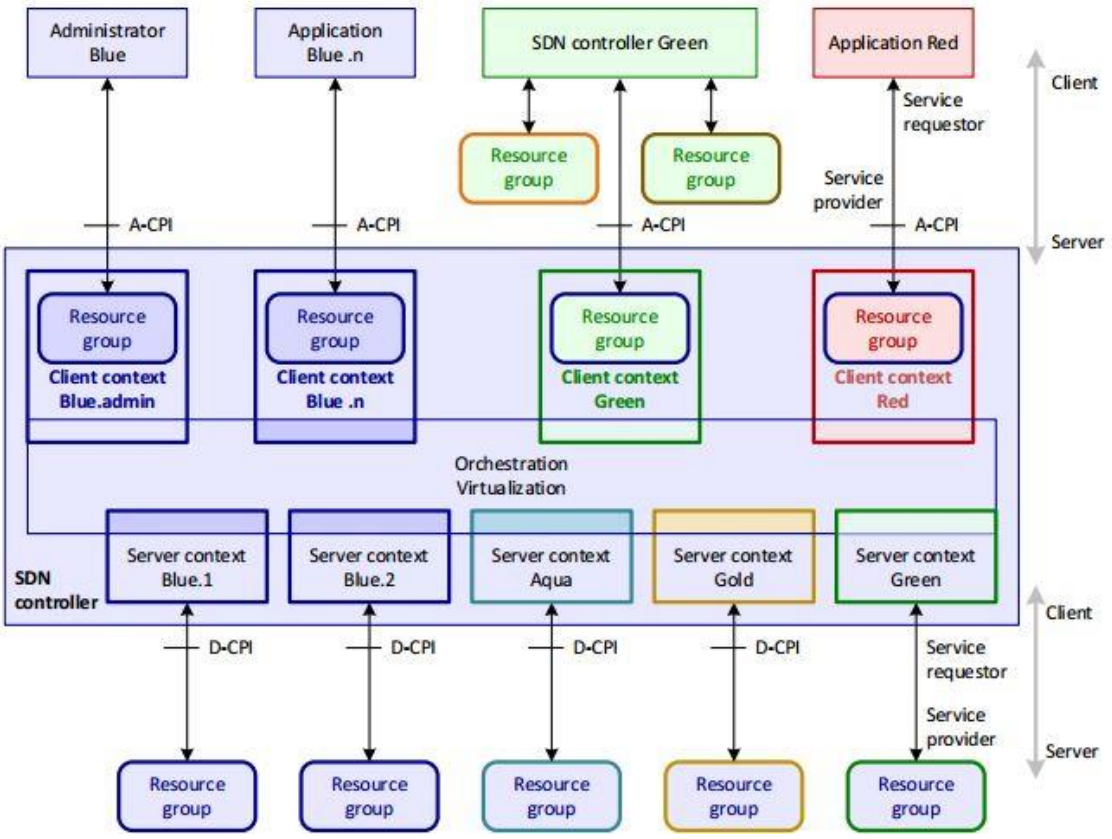


Figure 31: ONF SDN architecture extended overview [65]

ONF proposes here a **definition of orchestration** (even if defined only at technologies and resources CP level) as ongoing selection and use of resources by a server to satisfy client demands according to optimization criteria. In the implementation of orchestrator, two main abilities should be considered: Optimization algorithms and Policy to deal with complexity and real-time responsiveness to answer services SLA constraints of delay and jitter:

• **Policy** defined as an administrative rule or set of rules that specifies the action(s) to be taken when specified condition(s) occur, is very interesting for decision-making at the orchestrator level.

• **Optimization function** allows the orchestrator to adjust the resources under its control to a better optimum to answer the dynamic and changing clients/services requests.

Among services management life-cycle features, other significant orchestration functions are detailed in 9.4. The satisfaction of a client service request may require the orchestration function to select additional resources outside the pool available to the server. For instance, already assigned resources which are not used now, can be assigned to another client. The selection of such shared resources is affected by policy and resource traffic load or congestion. Because of virtual resources, service requests and notifications exist in specific client contexts. Therefore, the orchestration function must collaborate intimately with virtualization. The only

feature the entire community recognizes is the automated dynamically end-to-end (E2E) service delivery. All functions necessary to support it, may or may not be part of the orchestrator. However, in order to assure coherent management, keeping a holistic view of management remains mandatory. So, although ONF stays focused at the NFV and SDN levels and at open interface specifications, orchestration appears to be the heart of the SDN controller function.

In the last years, following the end of the specifications of the OpenFlow protocol, ONF directs now its efforts in platforms and project plans to deploy open-source solutions into their production networks. The objective is enabling a new supply chain ecosystem to help realize the full potential of SDN, disaggregation and open source [68]. Several POCs and field trials involve other open sources solutions such as open mobile evolved core (OMEC) (first full-featured, scalable, high performance open source Evolved Packet Core), ONOS or XaaS operating system (XOS) that are detailed in next sections (These open sources are under the umbrella of ONF.). To this end, ONF now releases both open-source components (ONOS, VOLTHA, Trellis, Stratum) and integrated platforms constructed from those components (ODTN, CORD, M-CORD, R-CORD, E-CORD).

9.1.2. Internet Engineering Task Force (IETF)

Internet Engineering Task Force (IETF) is originally more active in the LAN, Switches, and routers world. IETF was often involved in the specifications of simple and cost-effective protocols that have imposed themselves as de-facto standards, like SNMP over CMIP or Packets switching over ATM cells switching. This group usually works on protocol and interface specifications and less on general architectures. In this context, RFC-7426 [69] is more a general introduction to SDN. This document presents a general architecture and terminology with parallel CP and MP managing the resources and devices via an abstraction layer. Above this, the AP with the services and SDN applications (see Figure 32) control and use the CP and MP. This RFC does not specify any orchestration; the service decision fulfillment is done at the AP layer.

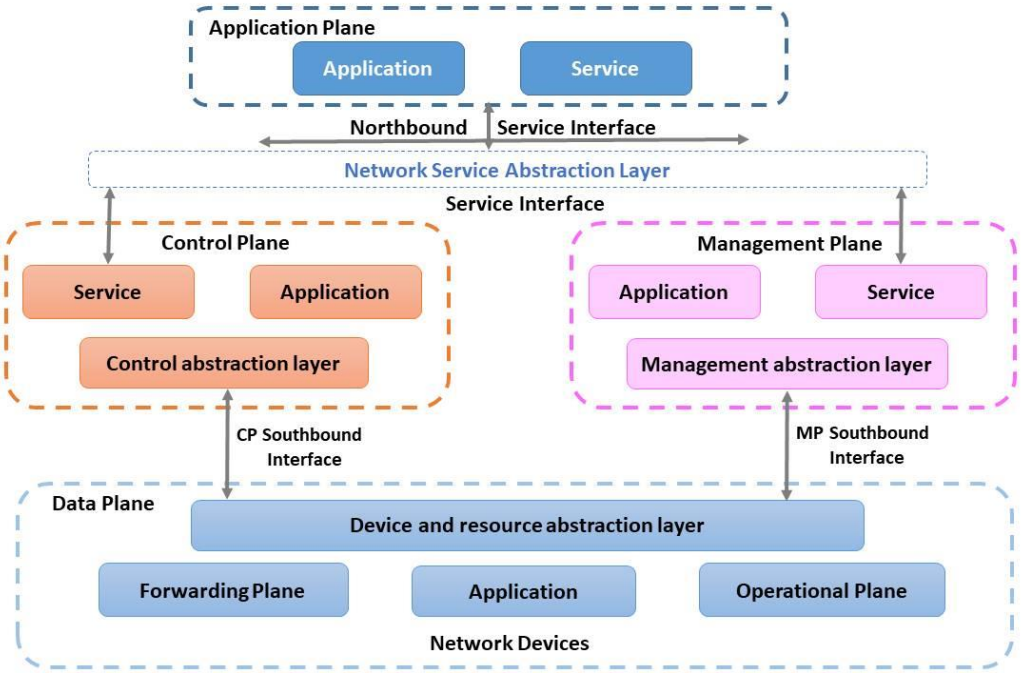


Figure 32: SDN layer architecture based on RFC-7426

Thus, in this architecture, emphasis is on main blocks and their different interfaces. CP functionalities usually include topology discovery and maintenance, packet route selection and instantiation, and path failover mechanisms. MP is more related to maintaining and monitoring network devices and their operational states. The main differences between CP and MP are timescale persistence and locality (CP more local and MP more centralized), and CP mostly introduces dynamicity. However, an East-West interface between CP and MP is missing. CP applications, such as resources optimization or services changes, may directly update the MP more efficiently. This RFC also presents succinctly protocols used at different levels of this architecture, such as ForCES, NETCONF/YANG, OpenFlow, I2RS, SNMP, PCEP, and bidirectional forwarding detection (BFD) for the CP and MP southbound layer; or CORBA, RPC, and REST for the Network Services abstraction layer.

In the RFC-6491, with the PCE-Based Architecture for application-based network operations (ABNO) [70, 71], IETF shows how some existing IETF components and protocols can be set together to make a complete SDN system with minimum changes and additions. ABNO appears as a functional and practical architecture without constraints to implementation choices. OSS/NMS applications can automatically provision network services and access network state information, with the help of a network policy agent (see Figure 33) without human intervention. In the same way, the OSS/NMS can issue high level services requests and interfaces with the policy agent, the traffic engineering database (TED) and label switched path database (LSP-DB), the operation administration and maintenance (OAM) handler, the application-layer traffic optimization (ALTO) server, and even the network devices. The application service coordinator

coordinates the activity of the network to provide services for use by applications like network as-a-service (NaaS) and complex services requests like virtual private network (VPN) or network slicing [72]. This application communicates with the ABNO controller to request operations on the network that can even come from the NMS/OSS. This Application can pre-qualify the services that will be requested to ABNO. ABNO Controller can be considered as a service orchestrator: it governs the behavior of the network in response to dynamic changing network conditions and in accordance with application network requirements and policies. It is the point of attachment and invokes the right components in the right order. The policy agent is responsible for propagating policies configured mainly by the OSS/NMS into the other components of the system and thus interfaces with most entities of this architecture, even with the interface to the routing system (I2RS) or with the path computation element (PCE) [73] that provides path computation and selection to the provisioning manager module. For implementation, multiple PCEs can operate on different TEDs in multi-domain or multi-layer networks. A domain in this case might be an autonomous system (AS), thus enabling inter-AS path computation. I2RS provides a programmatic way to access the routing state and policy information on routers in the network. The OAM handler is strategic and provides precious feedback to the NMS, OSS, controller, and administrator for fault and performances. It also allows a more flexible and dynamic behavior of the network. The virtual network topology manager (VNTM) is optional and can provision connectivity in the network physical layer, like configuring virtual links in optical wavelength-division multiplexing networks. A virtual network topology (VNT) is defined in RFC-5212 as a set of one or more label switched paths (LSPs) in one or more lower-layer networks that provides information for efficient path handling in an upper-layer network. Make-before-break for path test and selection is used for LSPs. The ABNO architecture includes a number of databases, such as TED and LSP-DB, but there may be more, containing information about topology (ALTO Server), policy (policy agent), or services templates at ABNO level. The ALTO server provides network information to the application layer based on abstract maps of a network region and allows path services optimization. Global concurrent optimization (GCO) [74], usually considered as part of NMS, is more dedicated to network optimization to avoid blocking problems and to achieve more optimal network-wide solutions. GCO can be used also in case of network failure.

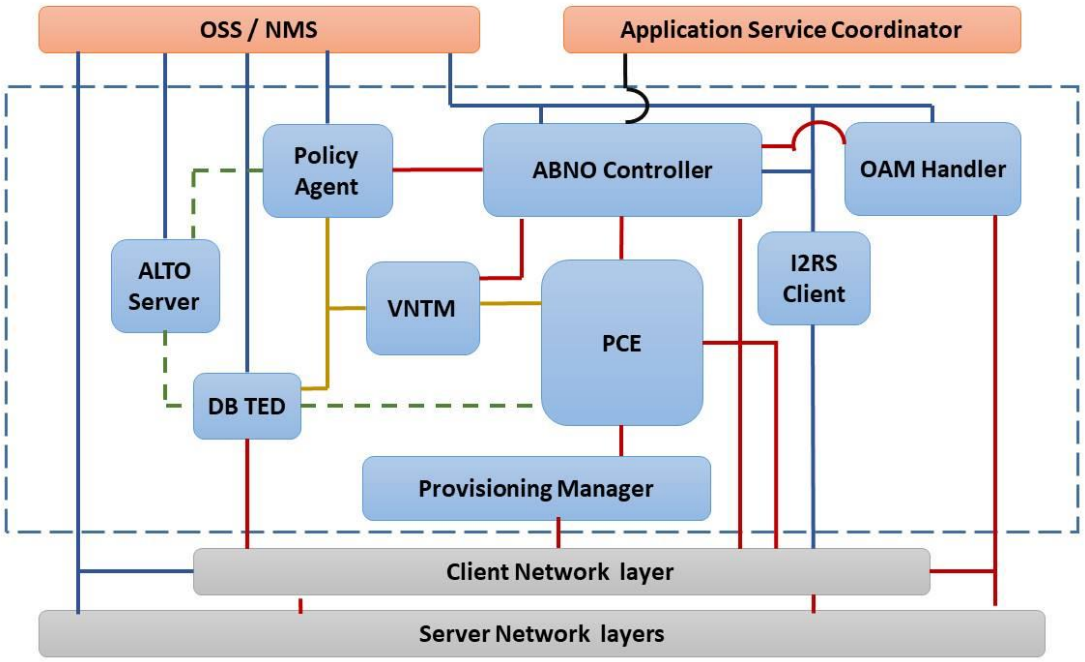


Figure 33: Generic ABNO architecture based on RFC-6491

Generally, IETF remains more preoccupied with protocol and interface specifications like NET-CONF/YANG (RFC-6241 and RFC-7950), and RESTful (RFC-8040). Implementations of ABNO already exist, such as iONE [75]. We can also cite a demonstration of SDN orchestration in optical multi-vendor scenarios, based on ABNO architecture with SDN solutions based on Ciena, Adva, Huawei, and Infinera, and validated with PCEP [26]. But these implementations stay at the POC level.

9.1.3. Metro Ethernet Forum (MEF)

Metro Ethernet Forum (MEF), mostly supported by vendors like Alcatel-Lucent, Huawei, Cisco, and AT&T, was founded in 2001 for support of layers 2 and 3 standard solutions, with specifications and certification processes. MEF now closes a gap at a higher layer of SDN architecture. The third network and lifecycle service orchestration (LSO) [76, 77, 24] sees the network as a NaaS for the end user and enables him to create, modify, and delete, dynamically and on-demand, services via customer Web portals or software (SW) applications. The forums aim is to replace the rigid OSS and BSS world and its silos approach [78] with horizontal orchestration layers in the newer SDN approach. As mentioned in [77], LSO is an agile approach to streamlining and automating the service lifecycle in a sustainable fashion for coordinated management and control across all network domains responsible for delivering an end-to-end connectivity service. Reference architecture (see Figure 34) describes the functional management entities needed to support LSO and the interfaces between them. LSO is related to orchestration and provides open and interoperable automation of management operations over the entire lifecycle of Layer 2 and 3 connectivity services. This includes design, fulfillment, control, testing, problem management,

quality management, billing & usage, security, analytics, and policy capabilities over the network domains, which require coordinated management and control in order to deliver the service. LSO MEF reference functional architecture is clearly defined with its limits and interfaces with other layers, (as shown in the figure below) and other partner domains. The LSO explains the different roles and functions of the customer application coordinator (CUS), of the business applications (BUS), and of the service orchestration functionality (SOF) within the same service provider (SP) domain and other partner domains and their respective interfaces. CANTATA, SONATA, ALLEGRO, and INTERLUDE are horizontal (east-west) APIs where LEGATO, PRESTO, ADAGIO are vertical (north-south) APIs. LSO MEF 55.1 defines complete sequence diagrams for product ordering and service activation orchestration or for controlling a service [24].

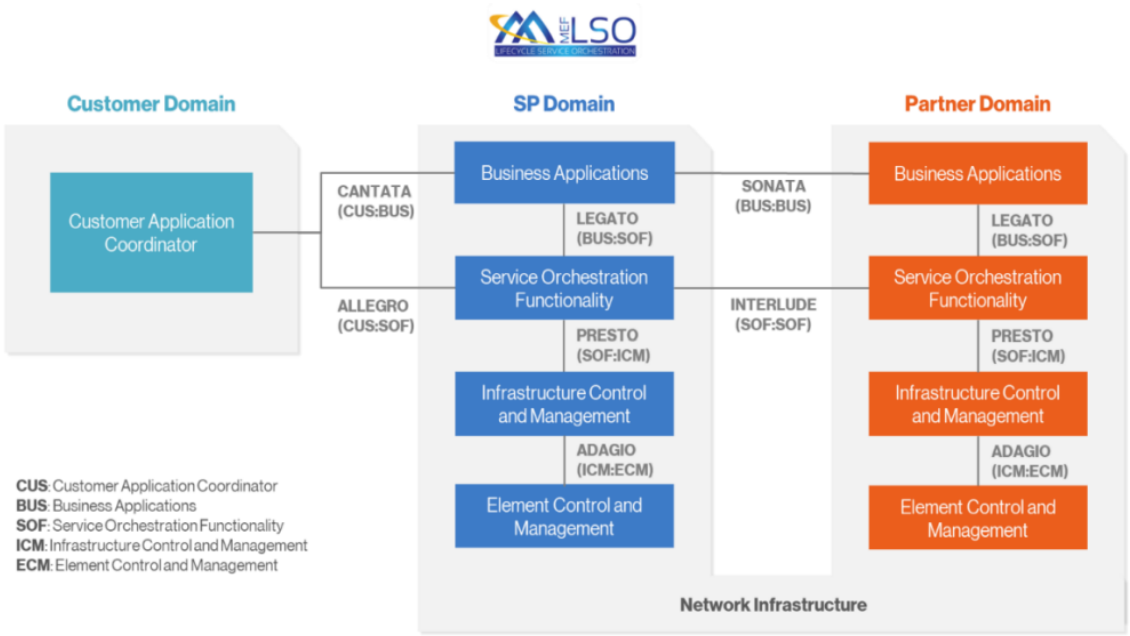


Figure 34: LSO reference architecture from LSO MEF [79]

MEF develops an open and standard architecture that should be able to support layer 2 and layer 3 services on demand with complete automation. The MEF unite program since 2014 coordinates internal and external engagement with standards development organizations (SDO) like ONF, ETSI, or TMF, Associations, and open-source projects like OPNFV (open NFV), ODL, or OpenStack to lead the industry migration to orchestrated services. Even though LSO MEF specifies every layer of the SDN architecture, it also gives preference to other forum partners like ONF or TMF for complete network support. Its objective is to define a detailed services orchestration via functional analysis and architecture for multi-domain management. From 2017, MEF focused on LSO API (application program interface) specifications. For instance, MEF and TM Forum announced they collaborate with major SPs to standardize LSO APIs for orchestrating connectivity services across multiple networks worldwide. Some of the largest SPs group their efforts within MEF to develop a complete suite of inter-provider LSO APIs that use the LSO reference architecture and the TM Forum open API framework around the LSO SONATA

reference point. LSO MEF also works on other API but essentially east-west like SONATA, INTERLUDE, CANTATA and ALLEGRO to accelerate interworking between companies. In June 2019, AT&T and Colt Technology Services implemented MEF's LSO Sonata APIs to automate network ordering [80]. The last two years, this MEF program is gaining importance. An LSO Sonata software development kit (SDK) has been implemented with APIs for inter-provider serviceability, product inventory, quoting, and ordering. This will allow a faster adoption of Sonata API. In the end of the year 2021, more than 23 service providers worldwide are in production with or involved with LSO APIs for automating inter-provider transactions. In October 2021, a new MEF 3.0 LSO API Certification Service is available, which includes the Sonata API. MEF also works with other forums like ONF to accelerate the development of interfaces between the control plane and the orchestration with specifications of Presto API [81]. It leverages ONF's Transport API model for network resource activation and topology, with the API integrated into an OpenDaylight SDN controller plug-in.

9.1.4. Tele Management Forum (TMF)

Tele management forum (TMF) is dedicated to telecom network management. Mainly located in the OSS, BSS, and network management areas [82], they now focus on the integration of SDN/NFV in order to present a general standard and open approach to future network management. According to TMF [18], orchestration definition is: E2E service management through zero-touch (automated) provisioning, configuration, and assurance across virtualized and physical components. The main reason for automating is to increase agility. In the proposed TMF SDN architecture [18], the role of the orchestration is identified at four different levels (see Figure 35): Customer management orchestration, Services orchestration, Resources management orchestration, and Technology management orchestration. TMF also uses MEF LSO as a reference for the four levels of orchestration.

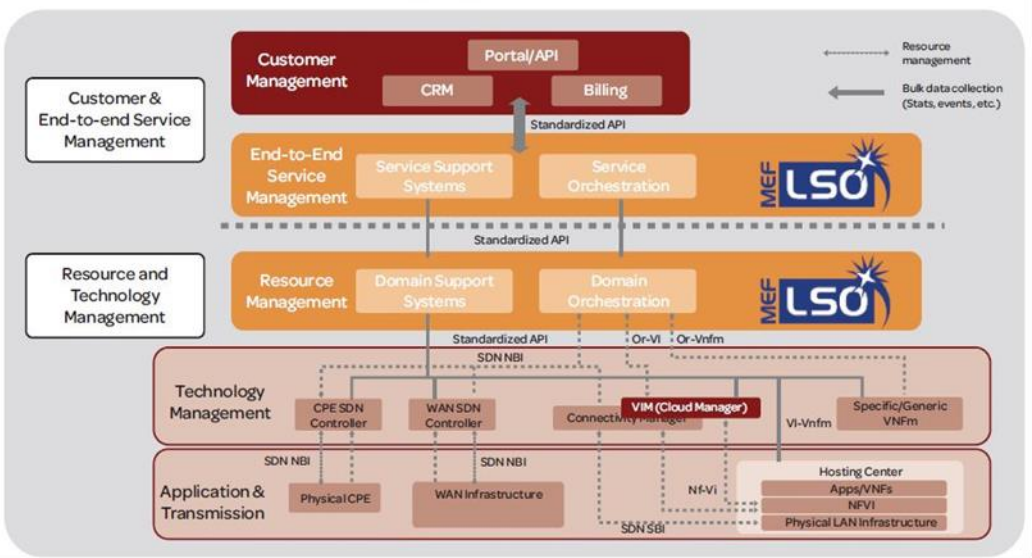


Figure 35: Architecture overview according to TMF Forum

Automation is required because of the different levels of abstraction, complexity, and latency requirements. For instance, applications driving 5G [83, 84] require 2 ms latency. The main reasons given by SPs for needing orchestration are as follows: faster delivery of new services (34% of SPs), ability to deliver services on demand with updates in real-time (28%), and reducing capital and operational expenditures (6%). These SPs define the following steps to achieve complete automation:

1. Transform infrastructure, back-office operations and business processes through NFV, SDN, and cloud technologies to become on-demand and efficient.

2. Turn those assets inside out as platforms that can serve partners and customers in a dynamic way and support radically new business models for themselves and third parties.

3. Make things orchestrable requires the following conditions:

(a) **CIM and common API patterns**, together with publication or catalog of the available services and their data to avoid vendor dependence. Standard APIs act as a bridge between an orchestration system and the OSS/BSS. Major international operators have officially adopted TM Forums suite of REST-based Open APIs for digital service management.

(b) **Automatic control loops and assurance** to enable rapid or even real-time response to requests for service. In networking, closing the loop means collecting and analyzing performance data to figure out how the network can be optimized and then applying policy, usually through orchestration, to make the changes in an automated way. Orchestration systems thus have to be data- and rule-driven.

(c) **Intent-based management** or declarative interfaces is the idea of abstracting the complexity of the network at a high level and then using intent and policy to manage it. The intent is what the customers want to do on the network, the service they want to use and the desired end state which they select through a self-service portal: I tell you what I intend to do, but not how to do it.

(d) **Permanent Service monitoring and assurance** to inform customers and refine policy if necessary. Moving to automated fulfillment is a huge step. So, you also have to know once the service is delivered if it was delivered correctly. Because of that, services analytics and machine learning for automatic services improvements are required (already implemented in British Telecom's own orchestrator). To do this, we need key performance indicators (KPIs) relevant to network performance, customer experience, and service quality in order to enable network changes, optimization, and self-healing.

(e) Planning clear technological and cultural **migration path** to a new management model, even if its priority is lower. Because these concepts are quite new, transition is still not the main preoccupation.

(f) **Security** should be designed from the beginning in order to protect the different servers' applications (more vulnerable to security breaches than proprietary telecom equipment networks).

These specific points are necessary to implement the operations center of the future (OpCF) [85, 86]. In this context, TM forums zero-touch orchestration, operations, and management (ZOOM) project works in order to develop the principles and guidelines to help SPs evolve their networks, support systems, and business processes to digital OpCF. Already, several catalyst POCs have been realized. In a recent TMF initiative, Orange and Huawei [87] have defined their vision of their future OSS when SDN, NFV, and orchestration are completely integrated. In sections 9.4 and 9.5, Table 11 and Table 12 expose the main features of the application orchestrator and the role of OSS as it is perceived in the future OSS.

Topology and orchestration specification for cloud applications (TOSCA), an orchestration data-modeling language managed by industry group OASIS describes the coordination between diverse resources across a potentially complex application environment. Both models TOSCA & YANG can work well together. For example, YANG can be used to define the interface for configuring individual VNFs, while TOSCA describes the E2E service including the creation, configuration, and chaining of VNFs. TOSCA is also ETSI MANO (see section 9.1.5) compatible.

Concerning the different SDN, NFV, and orchestration implementations (except the few described above), it is nearly impossible to cover every proposed solution. However, we can cite the following open sources implementations: Woodpecker version 2.6 last official release of **ONOS** (06/2021) is interoperable with the main open sources of the domain; ONOS has been designed for scalability, high performance, and high availability. Phosphorus fifteenth release (12/2021) of **OpenDayLight** (ODL) from Linux Foundation is an SDN, NFV platform, OpenFlow compatible. Phosphorus version supports Karaf Apache micro-services containers, is interoperable with open network automation platform (ONAP), and benefits of enhanced security. It includes also features such as WAN connectivity, Optical transport, enhanced cloud computing with improved support for network virtualization, Kubernetes micro-services and OpenStack support, or service function chaining (SFC). Updates to SFC accelerate delivery of services like network slicing, now supported by OpenvSwitch (OVS), allowing for improved adoption of SFC in the marketplace. After thirteen releases since 2013, ODL claims more than one billion subscribers today [88]. Beacon is a popular Java-based OpenFlow SDN controller. We can also cite OpenStack, Trema from NEC, or Ryu from NTT. OpenStack from Apache is the basis for several SDN applications; it is a free and open-source SW platform for cloud computing, mostly deployed as an IaaS. This SW platform consists of interrelated components that control diverse, multi-vendor hardware pools of processing, storage, and networking resources throughout a data center. Users either manage it through a web-based dashboard, through command-line tools, or through a RESTful API.

Hundreds of vendors SDN controllers are under development or already deployed and include basic features like network element discovery, identifying capabilities and gathering statistics and port states to allow basic control of the network. Some SDN controllers also have additional orchestration feature like decision-making rules or different network analytics tools. Almost all Telecom, IT, NMS, OSS, and BSS vendors have implemented their own solution. Among them, we can cite as SDN controllers: CISCO APIC; CISCO OSC; vendors version of ODL; HPE Distributed Cloud Networking (DCN); ECI LightCONTROL for multi-layer packet and optical networks; NOKIA NSP; Nuage network; and Red Hat NFV Platform based on Linux OpenStack. As orchestration platforms, we can cite: B4N from Brain4Net; Blue Planet from CIENA for service automation; CISCO XNC; Ericsson NFVi solution for 5G and IoT including NFV, SDN, and orchestration; Huawei Agile Controller 3.0 including service orchestration and based on ONOS; and Fujitsu Virtuora Service orchestrator based on ODL.

An important initiative, Central Office Re-architected as a Datacenter (CORD) [89], is supported by major operators and vendors like Google or CIENA. Its reference implementation, called POD, is built from hardware elements organized into a rack-able unit, commodity servers, and white-box switches, coupled with disaggregated access technologies (e.g., vOLT (optical line terminal), vBBU (baseband unit), vDOCSIS) and open-source SW (e.g., OpenStack, ONOS, XOS). XOS is a model-based platform, a kind of orchestrator, for assembling, composing, and managing services. It defines a service control plane that is layered on top of a diverse collection of mentioned above, back-end service implementations. Every solution implemented is usually proprietary (managing only its own product or technology), and most of them are based on previous open sources implementations.

9.1.5. European Telecommunications Standards Institute (ETSI)

European Telecommunications Standards Institute (ETSI) Industry Specification Group (ISG) NFV working group [90, 91], has played a significant role in the Mobile GSM standard and is still very active in the mobile world. ETSI answers the need to manage the Network Functions Virtualization Infrastructure (NFVI) created in the network virtualization context. This working group specifies the framework for NFV and Management and orchestration (NFV-MANO) of virtualized resources in the cloud data center that includes network elements, storage, and computing. ETSI MANO framework functional blocks can be separated into three main parts: The NFV architectural layers including NFVI and VNFs, the NFV MANO, and the network management part composed of EMS, NMS, OSS, and BSS, where every part is connected via a set of clearly defined interfaces. As displayed in Figure 36, NFV MANO is made of three entities:

1. NFV orchestrator (NFVO) is connected to Network Service (NS) and VNF catalogs taking into account new NS, NFV Instances, and NFVI Resources. NFVO supports life-cycle management of new NS including instantiation, scale-out/in, performance measurements, event correlation, and termination. Global resource management, validation, and authorization of network functions

virtualization infrastructure (NFVI) resource requests are also under its responsibility, together with policy management for NS instances.

2. VNF Manager (VNFM) supervises lifecycle management of VNF instances. It realizes the coordination and adaptation role for configuration and event reporting between NFVI and EMS/NMS.

3. Virtualized Infrastructure Manager (VIM) controls and manages the NFVI compute, storage, and network resources.

In more detail, NFV-MANO orchestration also includes additional features:

1. Testing role (for instance, performances testing of a service before providing it to a customer).

2. Open and standards interfaces with existing OSS/BSS to add dynamicity, speed, and agility for changing business needs and allowing personalized and easy to configure services to operators and end-users.

3. NFV policy management for rules management governing the behavior of NFV-MANO functions (e.g., management of VNF or NS scaling operations, access control, resource management, fault management, etc.). Policies are defined with conditions and corresponding actions. For example, a scaling policy may execute the related actions if the required conditions (e.g., VNF low CPU usage) were to occur during runtime. Different actions defined on the policy can be mutually exclusive, resulting in the process of selecting a particular action (or set of actions) to be executed. Once declared, a policy may be bound to one or more NS instances, VNF instances, and NFVI resources. NFV-MANO needs to support the execution of policies both automatically and manually (e.g., by requiring manual intervention from OSS/BSS).

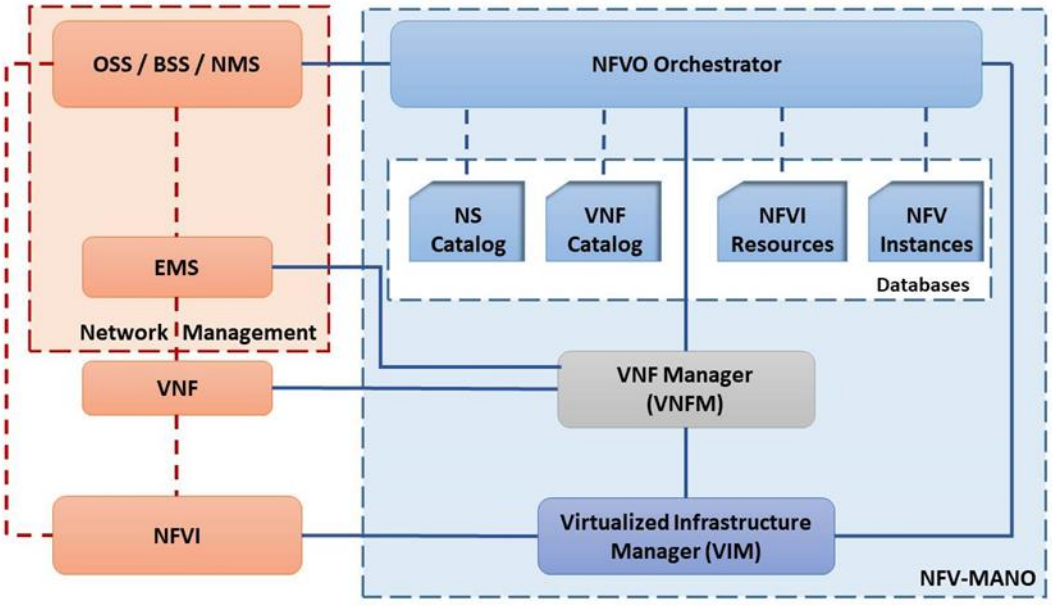


Figure 36: NFV MANO Architectural framework according to ETSI

To summarize, NFV-MANO architectural framework manages the NFVI and orchestrates the allocation of resources needed by NSs and VNFs. In this framework, open and standardized interfaces and API appear immediately as strategic to NFV-MANO success for management of internal or other administrative domains. Automation is also essential to assure real-time services and VNFs dynamic changes.

However, the MANO approach has a few limitations. With MANO-NFV, the NFVO does not know the meaning of the NFV it is managing, the NFV looks like a black box. In the MEF approach, LSO is aware of the physical resources and connection meaning of NFV. The OSS/BSS and EMS also maintain an important place and because of it, may allow a smoother transition to new model management with more flexible and dynamic services and resources management. However, ETSI MANO does not have a horizontal approach as does LSO MEF. According to reference [91], MANO still lacks details and standards for implementation for both manager applications and interfaces, which may cause interoperability issues.

In the last years, ETSI has specified the Experienced Network Intelligence (ENI) system [9], an innovative, policy-based, and model-driven functional entity that understands the configuration and takes actions in accordance with changes in context, such as the environment, the dynamic demand of the resources, and the varying service requirements. The aim is to enable intelligent service operation and management, and to enhance automated decisions taken by the system by using emerging technologies, such as big data analysis, artificial intelligence mechanisms, and also by automating complex human-dependent decision-making processes. This ENI system shall propose its services to the user, the application, the OSS/BSS, the orchestrator, or the infrastructure. This approach is different from the TMF [18] that has proposed the use of an orchestrator per layer to achieve similar goal. One complex ENI system that will be able to assist every module by proposed commands/recommendations, may allow to implement simpler orchestrators.

Among the several projects based on NFV-MANO framework, we can cite CloudNFV open platform, ExperiaSphere, and OpenMano open sources led by Telefonica. OPNFV with Iruya 9.0 current release, initiated and hosted by LINUX Foundation, works closely with ETSI and others to press for consistent implementation of open standards. OPNFV aims to be a carrier-grade, integrated platform that introduces faster new products and services to the industry. In 2015, Arno, the first outcome of OPNFV, implemented NFVI and VIM components. With Gambia, OPNFV has released its seventh version in Dec. 2019. TMF ZOOM [87] runs several Catalyst projects with NFV focus. Open-Source Mano (OSM) delivers an open-source MANO stack aligned with ETSI NFV Information Models. As an operator-led community, OSM is offering a production-quality open-source MANO stack that meets the requirements of commercial NFV networks. Several vendors commercial applications implemented by, among others, HP, Alcatel-Lucent, CISCO, and Cyan are also based on NFV-MANO.

Enhanced Control, orchestration, Management & Policy (ECOMP) [92] mainly initiated by AT&T is also important and coordinated with MEF. With ECOMP, ETSI MANO architecture is extended with Policy manager, controllers, and complete metadata specifications. ECOMP has been implemented to be the automation layer for its network and virtual functions. Open source from the beginning of 2017 with more than 8.5 million already written lines of code, this AT&T initiative participates significantly in MANO standardization. ECOMP was included in the second part of 2017, in ONAP [93] with its first version Amsterdam whose names better defines the real goals of autonomy and openness. ONAP presents the great advantage of being already partially deployed. Not only a model and data-driven approach to orchestration, Artificial Intelligence has been included too. One of the reasons AT&T has published its code for this tremendous project, is certainly because they hope to receive help from the Open-Source Community to achieve ONAP development, deployment, and adoption. ONAP attracts the interest of several major operators like Orange, Verizon, or China Mobile. Therefore, This project benefits from frequent versions rich in new features and well documented. The Beijing release standardizes and improves northbound interoperability for the ONAP Platform using the External API component and allows SDO collaborations, which are expected to support inter-operator exchanges and other use cases like vCPE or VoLTE defined by associated standards bodies such as MEF, TM Forum, and others. Casablanca version released in December 2018 [17], introduces new functionality with two use cases important to the evolution of networking: 5G and CCVPN (Cross Domain and Cross Layer VPN) with the use of MEF Cantata APIs. Casablanca also includes architectural changes, deploy-ability enhancements and improves its ability to collaborate with open sources projects such as OPNFV (Gambia version). Istanbul last version released in November 2021, and its previous versions El Alto and Honolulu, include mainly some security, stability, and policy enhancements, multi-level orchestration support, and enhanced E2E network slicing.

9.1.6. SDOs related work: summary

Several SDOs [GS1] contribute significantly to future NM at least at the vision level. Among them, a few worth citing are: The ONF forum is active at the lower NM layers and has specified the OpenFlow™ protocol [64, 65, 66]. SDN layer can control the network resources more easily and efficiently with the help of standard OpenFlow protocol. OpenFlow also has the advantage to cover several technologies like switches, radio, or optical network equipment. The IETF forum has aggregated several existing protocols to provide a rapid solution and deployment with complete network management architecture [69, 70, 71] but was barely used as basis for open sources. The MEF forum mostly invests at the E2E service level with the third network and LSO [76, 77]. MEF encourages active cooperation with forums such as ONF or TMF to converge to horizontal standard interfaces like Sonata between domains or vertical interfaces like Legato between Business applications and service Orchestration. This should allow on demand E2E service going over several ISPs. The TMF forum expresses the need for network and resource slicing for an automated cross-layer orchestration of those resources per domain, inter-domain, and cross-partners, depending on the business model along with cross-layer orchestration [18, 85, 86, 12]. The ETSI Institute has performed important work in the network function virtualization (NFV) domain [90, 91]. By decoupling service functions from infrastructures, NFV provides

benefits such as simplified and more flexible new services development and introduction and lower costs. With the ETSI specifications of the GS ENI, they also address the need for complex decision-making processes with the help of big data and AI. However, the SDN dimension is not clear enough, and EMS, NMS, and OSS still represent an important part of the network management. Open sources initiatives such as ONAP and CORD also propose a complete architecture implementation [17, 89] but remain at the level of big operators or important datacenters like Google. A careful reading reveals some gaps such as unclear frontier between the OSS and the orchestration or monolithic orchestration (except for TMF). They reveal also that the majority of these specifications and vision are still at the level of good will and will greatly influence the current NMS architecture. The aim of these future NM architectures is to satisfy every type of SR.

9.2. Research review

In the field of research concerning orchestration at the application level and OSS, we can distinguish three main directions: (1) the first one is more interested in technological aspects such as architecture, NFV, specific interface to orchestration, integration with cloud computing, or network slicing; (2) the second one has a more global approach and in the context of European projects, proposes broader solution for 5G, optical, satellite, or heterogenous networks. In the current context of complex networks and services, the aim is to attain autonomous (zero-touch) systems in the daily operation of the network. (3) the third direction has a more user and network centric approach: we examine the work realized in the domain of the service provisioning with its E2E and more local constraints. Among the multitude of projects, we refer to some of them in each category mostly where orchestration is considered. We are aware that we only cover a limited part of research in this area.

9.2.1. Research main concerns

In the research field, most of the new technologies are presented in Chapter 2. We just add here some interesting research concern that are related directly or indirectly to orchestration. The control orchestration protocol (COP) [94] allows an easier interface between SDN controllers and orchestration and proposes a common set of control plane functions used by various SDN controllers. COP can be compared to OpenFlow interface but at the level of heterogeneous control planes to allow a specific control plane interface per technology but also to provide a centralized orchestration functionality. COP can be compared to PRESTO API in the MEF LSO model. COP is defined using YANG and NETCONF despite it can be adapted to RESTconf. Projects of research such as STRAUSS, IDEALIST, or DISCUS carry out POC using COP.

As explained in section 2.8, network virtualization integrated with cloud computing [95] generates numerous academia and industry work where Key requirements for network-Cloud convergence include networking resource abstraction and exposure to upper layer applications and collaborations among heterogeneous systems across the networking and computing domains. Its architecture, based on Software Oriented Architecture (SOA) [96], allows loose coupling between

the services composed of network and computing resources. The network layer appears as a NaaS and this network-cloud convergence allows an enhanced service delivery and avoids the network to become a potential bottleneck in cloud computing services.

Some recent contributions focus on network management architecture. References [97, 19] present an SDN/NFV architecture for delivery of 5G and IoT services for multi-technological and multi-domain networks and the integration of SDN and NFV. The SDOs and research projects also contribute largely to this domain.

9.2.2. Projects in the research

Several research projects are already realized [26]. There are mostly European projects. Among them, we can cite the following:

T-NOVA NFV monitoring framework is a European project coordinated by the national center for scientific research "Demokritos" (EL) in Greece, with the participation of industrial and research companies like ATOS, HP, or Intel. T-NOVA has designed and implemented an integrated management architecture (network centric), including an orchestrator platform, for the automated provision, management, monitoring, and optimization of VNFs over network/IT infrastructures. T-NOVA was released as open source in May 2016 and is one of the first project covering NFV, including a Docker micro-service based NFV orchestration platform, called TeNOR [98].

Unify (https://www.eict.de/en/projects/) is a three years FP7 European project, that has been released also in 2016 including partners from university and industry. Its aim is to cover multiple technology domains to orchestrate joint network and cloud services concerning compute, storage, and networking. One interesting point of the UNIFY architecture is a general resource orchestrator connecting several resource orchestrators per technology controller. This project also includes several components such as VNF at L2 and L3 levels or orchestrator northbound interface for recursive and domain-oriented orchestration. At a lower level of the network management, several controller adapters southbound APIs have been implemented and tested in optical, radio, or data-center technology-specific domains. This allows to define SW programmability for every component of the network using NETCONF and YANG protocols for instance.

The research results of the T-NOVA and UNIFY projects have been utilized in a follow up project called **5G Exchange** from 2015 to 2018, whose objective is to enable cross-domain orchestration of services over multiple administrations.

SONATA, the service programming and orchestration for virtualized SW networks project [99], implemented from 2015 to 2017, comprises 15 important telecommunication partners and universities. SONATA was an EU-funded project Horizon 2020 and part of the 5G-PPP initiative. SONATA is an open-source project. It has implemented a SW development kit that supports functionalities and tools for the development and validation of VNFs and NS and a Service Platform, which offers the functionalities to orchestrate and manage network services during their lifecycles. This platform is ETSI NFV-MANO architecture compliant. The objective

of this project is to allow flexibility and deployment optimization of complex services and applications for 5G networks. Thus, SONATA is the first NFV integrated approach that included service composition, testing, and orchestration. This project has fulfilled three main objectives: (1) reduction of the time to market for networked services with the help of the SDK; (2) optimization of the resources and cost of service deployment and operation reduction, thanks to the Service Platform; and (3) acceleration of the adoption of NFV integration. SONATA intended to cover aspects in the cloud, SDN and NFV domains and to integrate network slicing in its platform. SONATA also contributed to SDOs such as ETSI, IETF, and ITU-T and to open-source solutions such as OSM and OpenStack. Research projects in the 5G domain had adopted SONATA platform in their implementations. 5GTANGO project is now enhancing and extending SONATA.

Implemented from 2017 to 2019, **5G-Transformer** project [100] also based on SONATA, comprises 18 actors from the industry and the academy. Its objective is to apply SDN/NFV/orchestration to transform the currently rigid mobile transport networks into a 5G dynamic system. The aim is to be able to offer network slicing to the specific needs of various vertical industries such as automotive, healthcare, and media/entertainment. For this purpose, 5GT defines 3 building blocks with the following innovations: (1) a vertical slicer for verticals to support the creation of their respective transport slices within a few minutes, (2) service orchestrator for end-to-end service orchestration and computing resources from multiple domains and manage their allocation to slices, and (3) mobile transport and computing platform. The E2E service orchestration supports multiple domains in a distributed way. One of its main outcomes is a full MANO stack featuring multiple advanced functionalities, such as vertical and network slice management and NFV network service composition and federation. Open source was also published in GitHub repository as well as many scientific articles and two Intellectual Property Rights have been registered. Several PoCs demos were performed such as a system for health emergencies based on 5G from 5TONIC laboratory together with SAMUR-PC and UC3M. Like SONATA, the proposed solutions are aligned with 3GPP and ETSI standards.

We can cite other research contributions like the **VITAL** project [101], which addresses the integration of terrestrial and satellite networks through the applicability of technologies such as SDN, NFV and multi-domains orchestration. We only cover few research projects that are often related to 5G networks. They propose interesting POC and contribute actively to orchestration, open-sources, and standard organizations such as ONF, ETSI, 3GPP, or TMF.

9.2.3. User and network centric approach

In current telecom networks, user and network centric approaches complete each other. The user centric approach where the user is at the center of the request is often more the concern of the operators who wants to simplify the management of the subscriber by allowing him to define his own services on demand by himself. As an example, we can cite Dapeng Wu [102], who proposes a user-centric 3C resource sharing model for Software-Defined ultra-dense network to lower the delay of the mobile cloud services. However, the majority of articles are more network centric and try to treat of the end-to-end service request considering all the different components

and links. Network management (NM) encounters several problems with respect to service request (SR) constraints at the application level and its E2E constraints, and at the network resource level, simultaneously. This function greatly affects the ability to set up an E2E SR and to modify the service dynamically. Numerous research focus on this problem. Some works emphasize the application E2E service constraints. In this area, particularly interesting investigation was done by Millnert [103]. This investigation uses mathematical models including input model, service model, and cost model, that consider the physical deployment of service elements in closed proximity in order to limit the E2E delay. Reference [104] proposes a mathematical model to optimize the number of machines needed for virtual network function (VNF) to guarantee the E2E constraints for network service (NS) setup by considering buffer and computing virtual machine (VM) constraints. However, there is no off-line procedure or scheduling model to determine the best E2E service and compare it with the chosen calculated one. In reference [105], the E2E Service Level Agreement (SLA) constraint is split in several local QoS constraints, reducing the complexity of the analysis. Such an approach improves the robustness of the schedule and allows analyzing each time-slice separately but does not warranty the respect of the E2E SLA. Reference [106] proposes a solution for Web services composition that consists of two steps: first, Mixed Integer Programming (MIP) is used to find the optimal decomposition of global QoS constraints into local constraints. Then, distributed local selection is used to find the best Web services that satisfy these local constraints. Here also service dynamicity is not covered.

Many other articles focus on improving local resources constraints such as [107], which propose to optimize the deployment and placement of the SDN controllers within a Wi-Fi network. T. H. Tan, et. al. [108] notably reduces the number of options for Web services at the level of local constraints by effectively discarding the service candidates that cannot satisfy global constraints. A. Sheoran et. al. [109] propose to limit the E2E delay constraint at the resource level by placing VNFs by affinity using micro-service aggregates within containers in the datacenter. These important contributions improve performance. However, they do not consider dynamic changes of user services including component modifications.

9.3. Limitations and Key features of the architectures of the different forums

As a general remark, among the different SDOs we have detailed before, there is a very large spectrum of architecture approaches and still a lack of convergence concerning the SDN orchestration architecture and functions. Table 9 illustrates this idea and summarizes the different levels of orchestration according to the forums and foundations detailed above. We can see in this table the missing convergence even if some synergies appear. For instance, MEF and TMF largely rely on ONF for the lower layers and may be thus complementary. The TMF forum has the most holistic approach based on its long history of network management, OSS, and BSS standardization. From TMF, we can expect a complete next-generation approach at the higher levels of management. Also, TMF will not have any problems working with lower layers based on ONF or ETSI approach.

Orchestration	IETF	ONF	MEF	TMF	ETSI
Customers orchestration	No	No	No	Yes[3]	No
Services orchestration	Yes[1]	No	Yes	Yes[3]	Yes[4]
Resources orchestration	No	Yes[2]	No	Yes[3]	No
Technologies orchestration	No	Yes[2]	No	Yes[3]	No

[1] At application layer in ABNO
[2] Specified only at the SDN controller
[3] Holistic NM approach
[4] With VNF and parallel to application layer

Table 9: Levels of orchestration according to the architectures of the different forums

What appears clearly from the first table, is that the orchestration is centered mainly on the service and resource layers. The orchestrator except the TMF forum, is seen as a monolithic application that has to address every request of the different modules, within modern network management.

In order to get an overview of the different forums and institutes, Table 10 summarizes some of strategic points of these architectures. This table is organized according to three main features of the future Network Management Systems (NMS): Dynamicity, Flexibility, and Adaptability [43] (see Figure 37).

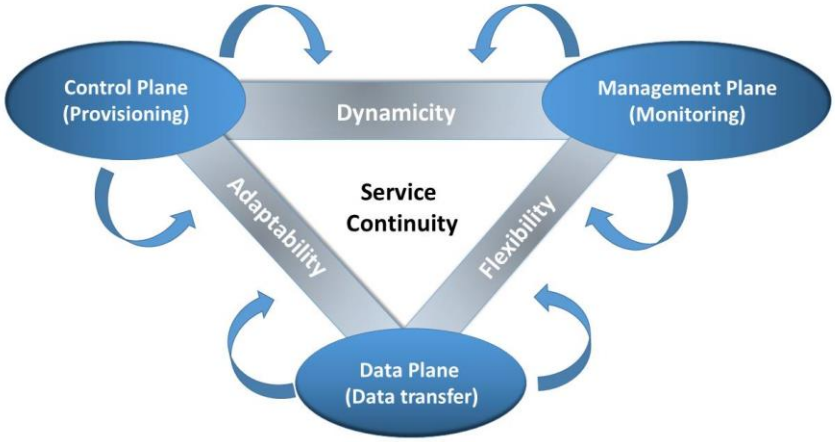


Figure 37: Convergence of the different planes

Regarding the three main planes of the network management, DP, CP, and MP, these features best define the collaboration and interaction between these planes in the NMS to allow their convergence:

1. **Dynamicity** well defines the relationship between the CP and MP. This is the ability to react in real-time to dynamic changes coming from the network, diverse VNFs, and customers with different and evolving service requests. For instance, with this feature, the two planes share monitoring information allowing them to work together, each one with its own responsibility but in complete integration and convergence. This collaboration is enforced by the orchestrator and additional virtualization at a higher level and can provide dynamicity that is hardly achieved today: for instance, the ability to choose an alternative

ubiquitous application server or to modify a service slice, without interrupting the service [10, 110] .

2. **Adaptability** is the result of the CP and DP collaboration and interaction. This illustrates the ability of the CP to continually take into account the user mobility, the different port states, or the bandwidth requests modifications, for instance with respect to the QoS and SLA.
3. MP and DP collaboration leads to **flexibility**. This is the ability of the operator to monitor, maintain, and bill differentiated network services using different types of SLA with the help of the dynamicity introduced by the interaction of CP and MP. For instance, this flexibility should allow personalized services like automatic switch to WIFI network when arriving at home.

Features	ONF Forum	IETF Forum	MEF LSO	TMF Forum	ETSI Institute
Flexibility: integrated mainly at the OSS level[1]	Yes, resource and technologies orchestration connected to OSS	Yes, several DB like OAM, ALTO, TED & even device level connected to OSS	At customer & business appl. with CANTATA, ALLEGRO & LEGATO	Future OSS vision: pre-order & order mgt in fulfillment entity through orchestration	EMS to VNFM and NMS, OSS to NFVO
Differentiated SLA network services	Yes, CP data and capability available to OSS	Yes, thanks to the different DBs	Yes, thanks to services orchestrator	Possible with closed control loops and autonomic operation	With data coming from virtualization VNFs, and NFVO
Reactive OSS/BSS[2]	Playing at DP, CP, and AP levels	Present above ABNO	Present above services orchestration	Place in the "future OSS vision" [89]	Present
Dynamicity: Network	CP with dynamic network states and services	Application service coordinator and ABNO	Mostly focused on services orchestrator and network dynamicity	Future OSS: merged MP & CP (horizontal approach)	Orchestration: dynamic services mgt via MANO
User service dynamicity	Not relevant	Not relevant	Want to allow on demand E2E service provisioning with multi-operators including user dynamicity	See NAAS for user self-service but limited to network dynamicity	At the network level only but including every component
From static to dynamic OSS	No, more services mgt in OSS – done at the CP layer	OSS configures policy agent, but no direct interface (I/F) with ABNO	Services under LSO responsibility	Application orchestration	Continuous work at BSS/OSS but no real CP
OSS features	Initial setup, policy config., PM, & SLA	Initial setup, policy config., PM, & SLA	New applications and engineering tools	Not relevant because of the integrated vision	Still important OSS role
Automation	Orchestration at the CP level	Limited automation at PCE level	At services orchestration	Zero-touch, future OSS (ZOOM)	Via NFVO and ENI entities
Adaptability: via open and standard I/Fs	OpenFlow, NetConf	OpenFlow, PCEP, NetConf, & ForCES between CP & DP[3]	Generic ADAGIO I/F	CIM, common API patterns, services catalog publication and their data[4]	No real CP but real importance of I/F
Virtualization	Possible resources virtualization between DP & CP	Virtual topology and links	Needed for services orchestration	Toward virtualized network	Focused on virtualization with VNF manager
Security:	At I/F and policy level, also during upgrade and SW download	Authorization and Authentication, Recommended inter-domain encryption	Strategic role based on authentication, encryption, trusted relationships[5]	Built-in security for the different servers' applications	Security group policies for access control Attention to Security elements to warranty entities portability

[1] EMS/NMS do not exist anymore in the different architectures except for ETSI.

[2] To state data, PM, and Accounting.

[3] RESTCONF and RESTful between CP and AP.

[4] MTNM, MTOSI at application Level.

[5] Authentication for all management interactions across LSO I/Fs, encryption across cross-administrative domain I/Fs, orchestrate the management of rule-based traffic filtering controls for connectivity services, and maintain info related to trust relationships with the domains and entities with which the components in LSO interact.

Table 10: Dynamicity, Flexibility, and Adaptability features in the NG architecture

A few remarks on Table 10 are necessary. ONF is now located at the first column of the table because it is not present at the application orchestration level. In the flexibility context, OSS should support rapid state data changes and not only model configuration or more frequent account reports.

Security feature has a central role. The SDN world is SW based, running on COTS servers in the cloud. Therefore, it is much more vulnerable than past telecom networks based on vendors' telecom equipment with often partially or completely proprietary interfaces. Authentication, authorization, and encryption are mentioned by every SDO. ONF and ETSI see security as built-in and include some security policies to be added even at upgrade or SW download. Authentication, encryption, and trusted relationships between operators appear to MEF as strategic functions to allow a secure E2E on demand service, going through several operators.

For the dynamicity feature, the SDOs are mostly focused on services and network dynamicity but are not concerned enough by the user growing needs for on demand dynamic services that can be ordered and modified without intervention of the operator. The MEF considering its important work at the multi-operator interoperability, is aware of this need to facilitate the creation of E2E services by users and their dynamicity even at the level of the VNFs composing the services.

Each of these architectures addresses the new requirements of the current network (5G, IoT) from a different entry point: ONF from the low levels interfaces and CP, the practical IETF with ABNO, reuse and integration of several existing entities with minimum additions and changes, MEF concerned with layer two and three E2E services provisioning and complete automation with services LSO, TMF with its holistic approach at higher layer and cultural migration path to new management model with future OSS, and ETSI with its focus on virtualization and AI. We can notice that several operators and companies are active simultaneously in several forums, open sources implementation, and private applications.

Despite missing convergence, several groups often complete each other and even build their contribution on the work realized by other forums. For instance, in multi-domain networks, ETSI domain management should be able to collaborate with another TMF managed domain via MEF CANTATA API as described in GS ENI 005 [9]. However, the main difficulties may be in the lack of clear interfaces and patterns specifications between the forums or even within different implementations based on the same architecture as ETSI MANO as explained before [91]. We can also observe that orchestration often appears as monolithic and is missing specifications and not yet clearly identified. The northbound interface especially is not clear and differs from forum to forum; mostly with the TMF future OSS that still needs consensus of the industry. This may also bring several interoperability conflicts.

Even if most vendors' SDN implementations are based on open sources largely supported by the industry, there are still a lot of private initiatives from start-ups, major hardware and software vendors, and ILECs. However, we observe recently that six Linux foundation open-source networking projects combine into one new project known as the Linux foundation networking fund (LFN) [111]. The initial projects OPNFV, OpenDaylight, FD.io, PDNA, and SNAS have recently integrated ONAP. This initiative will allow a work under a same umbrella, more cooperation, limit interoperability problems, and accelerate convergence. After the convergence of the TV, telephony, and data vendors, we now observe a new market concentration where main hardware and SW vendors like IBM, HP, Huawei, Nokia, or Alcatel-Lucent, are in

direct competition for this tremendous SDN market. The fact is, SDN and orchestration are already a necessity in the market, and forums do not have infinite time. These efforts are mostly visible in the WAN (wide area network). With the right synergies, all this together with programmable services, should evolve toward autonomous networks. Every-day work will be without human intervention, even if the deployed applications may quite differ from one operator to another.

In the domain research, several directions are proposed in order to converge to zero-touch network. They often focus on special points like provisioning, self-healing, or optimization based or inspired by an architecture of one of the forums. We have tried in this chapter to get a global view of the SDN, VNF and orchestration in order to shape and precise the functions and the organization of a complete integrated architecture. Among several research articles, we can note some interesting work on collaboration between the data-center and the application world usually related to SPs with the optical network world more related to network operators [110]. In the SUDOI proposed architecture, an interconnection and collaboration via SDN of the optical network and the data-center application (allocation of storage and computing resources) allows effective enhanced performances of the deployed services. Usually, the network and applications domains are managed separately except in the datacenters.

More generally in the research domain, with the introduction of VNF, the orchestration must synchronize the network path/slice and VNFs resources installed on virtual machines (also related to storage and computing resources allocation) to deploy an end-to-end service with QoS and delay respect [95]. We witness also a cloudification of the network that appears more and more as-a-Service (NaaS).

9.4. OSS/BSS evolution within the new architecture

The OSS cannot integrate the new features of the network described above because of its silos approach and because it was not created for this purpose. This is the reason the OSS role and features are evolving, as described in Table 11 below. This table is based on the previous chapters and retains a global approach. With the growing importance of the customer and application orchestrator, what features will be added to or removed from the OSS? What are the opportunities in terms of new applications?

OSS features	Features evolution in the OSS			Remarks
	Present	Transition	Future	
Fulfillment				
Qualified pre-order management	Yes	Partial	No Need (NN)	Supported in the service orchestration through complete automation even with NaaS – including service test before activation – although pre-order may first be done by the OSS: for example, for customer's privileges
Services activation and management	Yes	Partial	NN	Once the customer or the SP that requests the service is authenticated and eligible to receive it (according to its SLA)
Inventory: topology and services	Yes	Yes	NN	Will be done at SDN and service virtualization layers including new element discovery, topology, and services, and will serve upper applications and OSS for global operator usage
Engineering tools	Yes	Yes	Yes	Like planning tools, deployment tools, upgrades, or different backups at the elements, services, or DB levels
Orches. rules configuration	No	May	May	As part of OSS or just orchestrator configuration tool
Orches. services profiles DB	No	May	May	As part of OSS or just orchestrator configuration tool
Assurance				
Services management	Yes	Yes	Partial	For new features like network resource optimizations or energy saving, although sometimes seen at the resource and technologies management layer
Fault and event management	Yes	Yes	Partial	For long term network management and complex or multi-domains self-healing (based on **FMaaS** reports)
Performances and statistics	Yes	Yes	Partial	For new features like optimization or network extension requirements (based on **PMaaS**)
Workforce automation	Yes	Yes	Partial	For instance, trouble-ticketing will require human intervention when automatic self-healing is not enough
Probes systems	Yes	Yes	NN	At EMS/NMS and eventually at upper application layer

Table 11: Future OSS functions with integrated orchestration

In [87], TMF introduces an interesting concept: XaaS as described in the ONF part. For instance, as mentioned in Table 11, FMaaS or PMaaS applications collect their data from the network, from the CP and/or MP planes, and have a presentation layer. For instance, the performances of a network slice, a router, or a certain technology will be retrieved at the same interface of the PMaaS but filtered, depending on the needs of each. One can distinguish between network and hardware performances. This illustrates the idea of the convergence and integration of different planes (MP and CP in this case, each according to its specific needs) with the ability to assure an MCC.

In Table 11, an OSS picture of the future clearly appears. However, the transition path to this OSS will be long, painful, and expensive and first done at the WAN and clouds levels. It will depend on the existing networks and OSSs, network size, technologies used, and their respective

support for SDN and their interfaces, such as OpenFlow, NETCONF, or RESTFUL. There is no doubt that this transition is necessary; however, it will differ for every operator. The OSS will retain a global, strategic, and background role, which is especially significant for planning, financial, and marketing departments. The OSS remains the warrantor of the network and ensures its durability. Together with this, the OSS will not be able to integrate the dynamicity or to orchestrate the network because it was not initially created for this purpose.

The impact of the new architecture on the BSS, with its rating and billing, partners and interconnect, business optimization, and mediation applications, will be limited. It will consist mostly of integrating flexibility to be able to bill dynamic, mobile, and continuously changing services. For customer care, here also the main change will be the capability to integrate flexibility. Eventually [18], a kind of customer orchestrator may be added at the OSS level, allowing full use of the resources and services. Today already, some Web user portals allow customers to modify by themselves their rights according to their evolving needs. We even can imagine cloud-based application without any more need for SIM cards, where billing is based on different levels of rights to use the cloud-based network.

9.5. Application orchestration main features

In the previous section, we have seen the more global role of the OSS in the future, and its inability to respond to the dynamics or the orchestration of the network. So, we can now better identify the customer, service, and application orchestrator (CSAO) features as described in Table 12. In this table, we take out the ONF forum because it is mostly focused on the lower SDN layers. It is important to specify that the features listed below are the result of what is seen as important in the different forums. This does not mean that they are already implemented.

Features	IETF	MEF LSO	TMF	ETSI
OSS for long-term provisioning & assurance	No long-term vision	Yes, but via business layer	Yes, Zoom initiative with orches. Integrated to future OSS	Yes, with limited role
OSS and CSAO at the same layers	Same and above layers	Yes, with business appl. above	OSS / BSS above CSAO	At the same layer
Convergence for automation between OSS and CSAO[1]	Mandatory, appl. service coordinator as single I/F to ABNO for SR	Mandatory, to replace silos approach	Mandatory, with definition of next OSS generation	Mandatory
Services provisioning & life cycle management[1]	at ABNO and PCE answering the appl. service coordinator	Specified layers [2] and 3 lifecycle connectivity services	Specified	Specified at NFV orchestrator and also at OSS
Policy management: Decision-making rules	Mandatory, by policy agent configured by OSS	Mandatory	Mandatory, data- and rules-driven[2]	Mandatory, by policy management and ENI
Qualified notifications to different clients & subscribers	Mandatory, via OAM handler	Mandatory, with contextual or correlated events to related entities	Permanent service monitoring & assurance[3]	Published notifications of interest, according to applications needs
Performances for SLA respect[4]	Mandatory, via OAM handler	Mandatory, via PM metrics	Done via Visualization [89]	Supported in most MANO implementations
Performances for QoS for service assurance	No distinction between QoS and SLA PMs	Mandatory, via PM measurements	Via Operations: KPIs for network PM, customer experience, & Service QoS	For QoS measurement
Performances for billing & regulatory systems	Mandatory, with OSS and BSS I/F	Collection and reports of billing and usage	Mandatory, at customer management level	Not described
Network resource optimization	Supported by CGO application as part of the NMS	Not described	Mandatory, concurrent appl. continuously running[5]	Mandatory, by continuous work on resources
Path optimization	Within ALTO module providing network info	Not described	Mandatory[5]	Not mentioned
Testing function	LSP services test with help of OAM before validating	Critical role at the services orchestration level	Important, but time-consuming	Could be
- To recursively check resources availability for further usage	Described	Mandatory[6]	Automatic control loops	Recommended
Services & appl. requests coordination with other partner domains	Described via several ABNO domains (i.e., AS or inter-AS for PCE)	Described and partially specified with Sonata and Interlude interfaces	Done via orchestrators	Done at the NFV orchestrator level

[1]Automated dynamically E2E service delivery is mandatory for every forum.

[2]Machine learning for Policy support is recommended.

[3]To inform customers and refine policy if necessary.

[4]With resources attribution and service delivery (This includes State Information).

[5]Optimization algorithms are used and may be separate applications.

[6]In MEF, testing function is also mandatory to add flexibility to enable operational changes and variety.

Table 12: Application orchestration functionalities regarding OSS according to different SDOs

As a summary of this table, CSAO will be mostly involved in services or virtual functions provisioning, notifications, different types of performances reports, policy, and different kinds of

optimization and testing. CSAO is viewed as a significant part of the automation process. The CSAO configuration and its policy may be done at the OSS level, as a separate tool or at different levels of the architecture, as in ETSI.

In this table, we have differentiated between three types of performances: SLA, QoS, and billing. The first one allows services delivery and is required for services deployment. The second one continuously checks the delivered services with respect to the SLA and right QoS. The last one is dedicated to services billing. The distinction between SLA and QoS is not always clear in different forums.

We have also distinguished resources and path optimization, realized by two different applications at the CP and AP layers. Concerning the Fulfillment and Assurance, the features' repartition between OSS and CSAO appears here more clearly: CSAO is responsible for the network dynamicity due to user mobility and his constantly changing requirements, while the OSS gathers data for setup or sustainability of the network with a more global and static approach (thanks to the continuous monitoring). The transition to this repartition will be long and vary according to each operator's actual deployment. The question of open and standard interfaces between applications, OSS, and CSAO is essential, but also in the context of intra-domains collaboration. In ONF forum discussions, the services request between intra-domains should be considered at the service orchestration level and not at multi-domains or multi-technologies SDN controllers, as was thought at the beginning. For now, RESTful interface is often cited as application orchestrator northbound interface (NBI), but RESTConf and MTOSI are also considered for that.

9.6. Conclusion

Facing decreasing profits of the telecom operators and growing complexity of the network, every vendor and operator gets involved in the SDN revolution. After presenting the main SDN architectures in the major forums, and some main research concerns, we have focused on the place of the application orchestrator in each one. We have analyzed and compared these approaches according to three strategic features: dynamicity, adaptability, and flexibility within the different planes. We observe that all these forums, research, and implementations are still missing convergence, even when based on the same architecture. We have defined the role and place of the next-generation OSS, which will be more dedicated to global, strategic, and background tasks. With the help of this OSS analysis, we have defined the functions of the customers, services, and applications orchestration (CSAO) in order to attain programmable and zero-touch networks. CSAO will be involved in services and VNF life-cycle management, and in coordination, optimization, decision-making, and performance of the network. CSAO will have a complex and fundamental role in the future network management. In this context we have identified many challenges, such as standard APIs, central role of orchestration and its scalability, user-centric dynamicity, conflict management for decision-making, together with strict security requirements. In this context, the dynamicity and adaptability of the network appears as strategic features. Today SDN is very lively and perceived as the next step convergence between telecom SW and HW hardware vendors and the IT world; however, orchestration, which is still in its

infancy, presents several very stimulating challenges. Even if the different implementations and forums may appear quite confusing, initiatives like CORD, ONAP, and the work realized in the different forums and universities confirm this direction.

Chapter 10. Publications

[GS1] G. Saadon, Y. Haddad, N. Simoni, "A survey of Application orchestration and OSS in Next-Generation Network Management", Computer Standards & Interfaces, vol. 62, pp. 17-31, Fev. 2019.

[GS2] G. Saadon, Y. Haddad, N. Simoni, "Dynamic architecture based on network virtualization and distributed orchestration for management of autonomic network", In: 2019 15th International Conference on Network and Service Management (CNSM). IEEE, 2019; pp. 1-5.

[GS3] G. Saadon, Y. Haddad, N. Simoni, and M. Dreyfuss, "Optimal decision making in an SDN orchestrator", IET Networks, 11.1, pp. 13-26, 2022.

Chapter 11. Bibliography

[1] R. V. d. Meulen, "Gartner says 8.4 billion connected "things" will be in use in 2017, up 31 percent from 2016.," 7 February 2017. [Online]. Available: https://www.gartner.com/en/newsroom/press-releases/2017-02-07-gartner-says-8-billion-connected-things-will-be-in-use-in-2017-up-31-percent-from-2016.

[2] I. Markit, "More than six billion smartphones by 2020, ihs markit says," IHS Markit, 18 January 2017. [Online]. Available: https://technology.informa.com/587841/more-than-six-billion-smartphones-by-2020-ihs-markit-says.

[3] L. S. Vailshery, "IoT connected devices worldwide 2030," statista, 22 January 2021. [Online]. Available: https://www.statista.com/statistics/802690/worldwide-connected-devices-by-access-technology/. [Accessed 25 January 2022].

[4] F. Barbarulo, "Extending ETSI MEC toward stateful application relocation based on container migration," in *23rd International Symposium on a world of Wireless Mobile and Multimedia networks (WoWMoM)*, 20222.

[5] P. S. Shenker, "SDN course," 2014. [Online]. Available: https://player.slideplayer.com/33/8230322/#.

[6] A. S. Thyagaturu, A. Mercian, M. P. McGarry, M. Reisslein and W. Kellerer, "Software defined optical networks (SDONs): a comprehensive survey," *IEEE Communications Surveys & Tutorials,* vol. 18, no. 4, p. 2738–2786, 2016.

[7] "Microwave information model," Tech. Report, TR-532 V1.0, ONF, Dec. 2016.

[8] S. Sasidharan and S. K. Chandra, "Defining future SDN based network management systems characterization and approach," in *Computing, Communication and Networking Technologies (ICCCNT), 2014 International Conference on, IEEE*, Jul. 2014.

[9] *ETSI GS ENI 005 V2.1.1, Experiential Networked Intelligence (ENI), System Architecture,* ETSI, 2021.

[10] Q. Duan and S. Wang, Network As a Service for Next Generation Internet, IET Book, 2017.

[11] A. Neal, "Study on New Services and Markets Technology Enablers," 3GPP – TR 22.891 rel. 14.2, Sep. 2016.

[12] T. Ben Meriem, T. Nakamura, S. Horiuchi, S. Fratini, D. Milham and A. Pope, "A Dynamic Network Slices Management and Business Models, rel. 17.0.1," tmforum, Nov. 2017.

[13] W. Knight, "The dark secret at the heart of AI, MIT Technology review," Apr. 2017. [Online]. Available: https://www.technologyreview.com/2017/04/11/5113/the-dark-secret-at-the-heart-of-ai/.

[14] J. Gao, V. Gruhn, J. He, G. Roussos, W.-T. Tsai and et al., "Mobile cloud computing research-issues, challenges and needs," in *", Service Oriented System Engineering (SOSE), 2013 IEEE 7th International Symposium on*, 2103.

[15] Z. Sanaei, S. Abolfazli, A. Gani and M. Shiraz, "Sami: Service-based arbitrated multi-tier infrastructure for mobile cloud computing," in *Communications in China Workshops (ICCC), 2012 1st675 IEEE International Conference on, IEEE*, 2012.

[16] T. Cagenius and et al., "Evolving the TV experience: Anytime, anywhere, any device," Ericsson Review No. 3, 2006.

[17] "Open Network Automation Platform (ONAP) Architecture, White Paper Casablanca release," The Linux Foundation, Nov. 2018.

[18] D. Bushaus, A. Turner, R. Rich, S. Wray and P. Davis, "Orchestration, get ready for the platform revolution," report, ISBN: 978-1-945220-04-3, tmforum, Sep. 2016.

[19] Q. Duan, N. Ansari and M. Toy, "Software-Defined Network Virtualization: An Architectural Framework for Integrating SDN and NFV for Service Provisioning in Future Networks," *IEEE Network Magazine,* vol. 30, no. 5, pp. 10-16, Sep. 2016.

[20] P. Mell and T. Grance, "The NIST definition of cloud computing," United States, 2011.

[21] H. Cai, K. Zhang, M. Wang, J. Li, L. Sun and X. Mao, "Customer centric cloud service model and a case study on commerce as a service," in *International conference on cloud computing, IEEE, pp. 57-64*, 2009.

[22] A. Erradi, P. Maheshwari and V. Tosic, "Policy-driven middleware for self-adaptation of web services compositions," in *ACM/IFIP/USENIX International Conference on Distributed Systems Platforms and Open Distributed Processing, pp. 62-80*, Springer, Berlin, Heidelberg, Nov. 2006.

[23] E. Marilly, O. Martinot, H. Papini and D. Goderis, "SLA: a main challenge for Next Generation Networks," Alcatel IEEE, Aug. 2002.

[24] M. standard, *Lifecycle Service Orchestration (LSO): Reference Architecture and Framework, MEF 55.1,* MEF, Jan. 2021.

[25] "ETSI GS Group Specifications ENI 001 V3.1.1," 12 2020. [Online]. Available: https://www.etsi.org/deliver/etsi_gs/ENI/001_099/001/03.01.01_60/gs_ENI001v030101p.pdf. [Accessed 01 2022].

[26] N. F. S. de Sousa, D. A. L. Perez, R. V. Rosa, M. A. Santos and C. E. Rothenberg, "Network service orchestration: A survey," *Computer Communications, Elsevier,* Vols. 142-143, pp. 69-94, Jun. 2019.

[27] A. Osseiran, F. Boccardi, V. Braun, K. Kusume, P. Marsch, M. Maternia, O. Queseth, M. Schellmann, H. Schotten, H. Taoka and et al., "Scenarios for 5g mobile and wireless communications: the vision of the metis project," *IEEE communications magazine,* vol. 5, pp. 26-35, 2014.

[28] M. S. Ali, E. Hossain and D. I. Kim, "LTE/LTE-a random access for massive machine-type communications in smart cities," *IEEE Communications Magazine,* vol. 55, no. 1, pp. 76-83, 2017.

[29] X. Wang, D. Lan, X. Fang, M. Ye and Y. Chen, "A resource management frame work for multi-tier service delivery in autonomic virtualized environments," in *NOMS 2008- 2008 IEEE Network Operations and Management Symposium, pp. 310–316*, 2008.

[30] S. Scott-Hayward and E. Garcia-Palacios, "Multimedia resource allocation in mmwave 5g networks," *IEEE Communications Magazine,* vol. 53, no. 1, pp. 240-247, 2015.

[31] A. Tanenbaum and D. Wetherall, Computer networks, chap. 1 and chap. 5 pp. 362–389, 5th Edition, Cloth: Prentice Hall; pearson, 2011.

[32] S. Chen, F. Qin, B. Hu, X. Li and Z. Chen, "User-centric ultra-dense networks for 5g: challenges, methodologies, and directions," *IEEE Wireless Communications,* vol. 23, no. 2, pp. 78-85, 2016.

[33] J. Kosinski, P. Nawrocki, D. Radziszowski, K. Zielinski, S. Zielinski, G. Przybylski and P. Wnek, "SLA monitoring and management framework for telecommunication services," in *Fourth International Conference on Networking and Services (icns 2008), IEEE, pp. 170-175*, 2008.

[34] V. Casola, A. Mazzeo, N. Mazzocca and M. Rak, "A SLA evaluation methodology in

service-oriented architectures," in *Quality of Protection, Springer, pp. 119–130*, 2006.

[35] K. B. Costa and e. al, "Enhancing orchestration and infrastructure programmability," *IEEE Access,* vol. 8, p. 195487–195502, 2020.

[36] F. M., G. M., F. M., K. R. and U. M., "Automatic orchestration of video streams to enhance group communication," in *international workshop on Socially-aware multimedia*, Madrid, Spain, 2012.

[37] C. Sonmez, A. Ozgovde and E. C., "Fuzzy workload orchestration for edge computing," *IEEE Transactions on Network and Service Management,* vol. 16, no. 2, p. 769–782, 2019.

[38] l. Lammers and T. Hoppe, "Watt rules? assessing decision-making practices on smart energy systems in dutch city districts," *Energy research social science,* vol. 47, p. 233–246, 2019.

[39] A. Dahbi, Y. Balouki and T. Gadi, "The selection of the relevant association rules using the electre method with multiple criteria," *IAES International Journal of Artificial Intelligence,* vol. 9, no. 4, pp. 638-645, 2020.

[40] M. Dreyfuss and M. Sher, "Policies for operating enforcement cameras," *Journal of Transportation Safety & Security,* vol. 12, no. 6, pp. 746-763, 2020.

[41] A. Guyton, Textbook of Medical Physiology 8th Ed., AC Guyton & JE Hall, eds, Harcourt College Publishers, 1990.

[42] N. Huin, B. Jaumard and F. Giroire, "Optimal network service chain provisioning," *IEEE/ACM Transactions on Networking,* vol. 26, no. 3, pp. 1320-1333, 2018.

[43] K. Qiu, J. Zhao, X. Wang, X. Fu and S. Secci, "Efficient recovery path computation for fast reroute in large-scale software-defined networks," *IEEE Journal on Selected Areas in Communications,* vol. 37, no. 8, pp. 1755-1768, 2019.

[44] M. Potamias, F. Bonchi, C. Castillo and A. Gionis, "Fast shortest path distance estimation in large networks," in *Proceedings of the 18th ACM conference on Information and knowledge management, pp. 867–876*, 2009.

[45] A. Boubendir, "Flexibility and dynamicity for open network-as-a-service from architecture modeling to deployment," phdthesis, Telecom ParisTech, Mar. 2017.

[46] G. e. a. SUN, "Energy-efficient and traffic-aware service function chaining orchestration in multi-domain networks," *Future Generation Computer Systems,* vol. 91, pp. 347-360, 2019.

[47] X. Yang, D. Lo, X. Xia, Y. Zhang and J. Sun, "Deep learning for just-intime defect prediction," in *2015 IEEE International Conference on Software Quality, Reliability and*

Security, pp. 17–26, 2015.

[48] L. Gupta, M. Samaka, R. Jain, A. Erbad, D. Bhamare and H. A. Chan, "Fault and performance management in multi-cloud based NFV using shallow and deep predictive structures," in *2017 26th International Conference on Computer Communication and Networks (ICCCN), pp. 1-8*, 2017.

[49] A. Javaid, Q. Niyaz, W. Sun and M. Alam, "A deep learning approach for network intrusion detection system," in *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS), pp. 21–26*, 2016.

[50] S. Ayoubi, N. Limam, M. A. Salahuddin, N. Shahriar, R. Boutaba, F. Estrada-Solano and O. M. Caicedo, "Machine learning for cognitive network management," *IEEE Communications Magazine,* vol. 56, no. 1, pp. 158-165, 2018.

[51] C. Jiang, H. Zhang, Y. Ren, Z. Han, K.-C. Chen and L. Hanzo, "Machine learning paradigms for next-generation wireless networks," *IEEE Wireless Communications,* vol. 24, no. 2, pp. 98-105, 2016.

[52] "Openflow switch specification," Technical Specifications, TS-025, ONF, Mar. 2015.

[53] "Wireless transport SDN proof of concept white paper," White Paper, ONF, Oct. 2015.

[54] "Wireless transport SDN proof of concept 2 detailed report," ONF White Paper, Jun. 2016.

[55] R. Vilalta, A. Mayoral, J. Baranda, J. Nunez, R. Casellas, R. Martınez, J. Mangues-Bafalluy and R. Muñoz, "Hierarchical SDN orchestration of wireless and optical networks with E2E provisioning and recovery for future 5G networks," in *Optical Fiber Communications Conference and Exhibition (OFC), IEEE, pp. 1-3*, 2016.

[56] O. Aliu, C. Niephaus and M. Kretschmer, "Configuration of the wireless interface for software defined wireless networks," in *NetSoft Conference and Workshops (NetSoft), IEEE, pp. 166–170*, 2016.

[57] D. Bercovich, L. M. Contreras, Y. Haddad, A. Adam and C. J. Bernardos, "Software-defined wireless transport networks for flexible mobile backhaul in 5G systems," *Mobile Networks and Applications, Springer,* vol. 20, no. 6, pp. 793-801, Dec. 2015.

[58] C. Q. and et al., "Functional requirements for transport API," Tech. Report, TR-527, ONF, Jun. 2016.

[59] A. Jajszczyk, A Guide to the Wireless Engineering Body of Knowledge (WEBOK), 2nd Edition, Wiley-IEEE Press, Nov. 2012.

[60] J. S. Bellotti and F. Gruman, "Optical Transport Protocol, Extensions, Version 1.0," ONF TS-022, Apr. 2017.

[61] N. D. K. Lam, Core information model, fragments of the ONF common information model, TR-512 version 1.1, ONF, Nov. 2015.

[62] T. Yamaguchi, T. Sato, H. Takeshita, S. Okamoto and N. Yamanaka, "Experimental report of elastic lambda aggregation network control method for SDN-based carrier class network," in *Optical Internet 2014 (COIN), 2014 12th International Conference on, pp. 1-2, IEEE*, 2014.

[63] W. Cheng and L. Wang, "MPLS-TP OpenFlow Protocol, Extensions for SPTN, Version 1.0," ONF TS-029, Jun. 2017.

[64] ""SDN architecture overview", Technical reference, TR-504 version 1.1," ONF, 2014 November.

[65] "SDN architecture issue 1.1," Technical Reference, TR-521, ONF, Feb. 2016.

[66] "Orchestration: A more holistic view," Technical Recommendation, TR-540, ONF, Jan. 2017.

[67] V. Lopez, L. M. Contreras, O. G. de Dios and J. P. F. Palacios, "Towards a transport SDN for carriers networks: An evolutionary perspective," in *Networks and Optical Communications (NOC), 2016 21st European Conference on, IEEE*, Jul. 2016.

[68] "ONF Strategic Plan," ONF Board, Mar. 2018.

[69] E. Haleplidis, K. Pentikousis, S. Denazis, J. H. Salim, D. Meyer and O. Koufopavlou, "Software-Defined Networking (SDN): Layers and architecture terminology," Tech. Report, RFC 7426, IRTF, Jan. 2015.

[70] C. Rotsos, D. King, A. Farshad, J. Bird, L. Fawcett, N. Georgalas, M. Gunkel, K. Shiomoto, A. Wang, A. Mauthe and et al, "Network service orchestration standardization: a technology survey," *Computer Standards & Interfaces,* vol. 54, pp. 203-215, Nov. 2017.

[71] D. King and A. Farrel, "A PCE-based architecture for application-based network operations," Tech. Report RFC-7491, IETF, Mar. 2015.

[72] I. T. Haque and N. Abu-Ghazaleh, "Wireless software defined networking: a survey and taxonomy," *IEEE Communications Surveys & Tutorials,* vol. 18, no. 4, p. 2713–2737, 2016.

[73] A. Farrel, J.-P. Vasseur and J. Ash, "A Path Computation Element (PCE)-based architecture," Tech. Report, RFC-4655, IETF, Aug. 2006.

[74] Y. Lee, J. Roux, D. King and E. Oki, "Path Computation Element Communication Protocol (PCEP) requirements and protocol extensions in support of global concurrent optimization," Tech. Report, RFC-5557, IETF, Jul. 2009.

[75] L. Velasco and L. Gifre, "ione: A workflow-oriented ABNO implementation," in *Photonics in Switching (PS), 2015 International Conference on, pp. 330–332, IEEE*, 2015.

[76] A. Mayer and S. Mansfield, "The third network: Lifecycle service orchestration vision," white paper, MEF, Feb. 2015.

[77] "Lifecycle service orchestration (LSO): Reference architecture and framework," service operations specification, MEF 55, Mar. 2016.

[78] "Principles for the management of next generation networks," ITU-T recommendation M.3060/Y.2401, ITU, Mar. 2006.

[79] V. Lopez, L. Miguel, J. Foster, H. Silva, L. Blair, J. Marsella, T. Szyrkowiec, A. Autenrieth, C. Liou, A. Sadasivarao and et al., "Demonstration of SDN orchestration in optical multi-vendor scenarios," in *Optical Fiber Communication Conference and Exhibition (OFC), pp. 1-3, IEEE*, Mar. 2015.

[80] J. Lyons Hardcastle, "AT&T, Colt Use MEF's LSO Sonata APIs to Automate Network Ordering, SDxCentral," Jun. 2019. [Online]. Available: https://www.sdxcentral.com/articles/news/att-colt-use-mefs-lso-sonata-apis-to-automate-network-ordering/2019/06/.

[81] "Network Resource Provisioning, Interface Profile Specification," MEF 60, Jan. 2018.

[82] "Enhanced Telecom Operations Map (eTOM), standard," tmforum, Feb. 2007.

[83] "The 5G-infrastructure public private partnership.," 2020. [Online]. Available: https://5g-ppp.eu/.

[84] R. Chavez-Santiago, M. Szydelko, A. Kliks, F. Foukalas, Y. Haddad, K. E. Nolan, M. Kelly, M. Masonta and I. Balasingham, "5G: The convergence of wireless communications," *Wireless Personal Communications,* vol. 83, p. 1617–1642, Aug. 2015.

[85] N. David and T. Degroot, "OSS / BSS futures: Preparing the future mode of operation, Tech. Report 2.0.3," tmforum, Mar. 2016.

[86] D. Bushaus, S. Wray and P. Davis, "Agile operations: Moving towards the operations center of the future, Report," tmforum, Mar. 2016.

[87] "Future OSS: Providing the agility to support digital operations transformation of hybrid networks, White Paper Rev. 1.0," Huawei and Orange, Feb. 2017.

[88] J. Lovato, "OpenDaylight, Most Pervasive Open Source SDN Controller, Celebrates Sixth Anniversary with Neon Release," The Linux Foundation, Feb. 2017. [Online]. Available: https://www.linuxfoundation.org/press-release/2019/03/opendaylight-most-pervasive-open-source-sdn-controller-celebrates-sixth-anniversary-with-neon-release/.

[89] L. Peterson, A. Al-Shabibi, T. Anshutz, S. Baker, A. Bavier, S. Das, J. Hart, G. Palukar and W. Snow, "Central office re-architected as a data center," *IEEE Communications Magazine,* vol. 54, no. 10, pp. 96-101, Oct. 2016.

[90] R. Mijumbi, "Network Functions Virtualization (NFV); management and orchestration, Tech. Report," ETSI GS NFV-MAN 001 V1.1.1, ETSI, Dec. 2014.

[91] R. Mijumbi, J. Serrat, J.-I. Gorricho, S. Latre, M. Charalambides and D. Lopez, "Management and orchestration challenges in network functions virtualization," *IEEE Communications Magazine,* vol. 54, no. 10, pp. 98-105, 2016.

[92] "ECOMP (enhanced control, orchestration, management & policy) architecture. White paper," AT&T, 2016.

[93] "ONAP delivers Amsterdam release sets standard network service automation," ONAP, Nov. 2017.

[94] R. Vilalta and et al., "The need for a control orchestration protocol in research projects on optical networking," in *2015 European Conference on Networks and Communications (EuCNC), IEEE*, 2015.

[95] Q. Duan, Y. Yan and A. Vasilakos, "A survey on service-oriented network virtualization toward convergence of networking and cloud computing," *IEEE Transactions on Network and Service Management,* vol. 9, no. 4, pp. 373-392, 2012.

[96] K. Channabasavaiah, K. Holley and E. Tuggle, "Migrating to a Service Oriented Architecture," IBM, Apr. 2004.

[97] R. Vilalta, A. Mayoral, R. Casellas and et al., "SDN/NFV orchestration of multi-technology and multi-domain networks in cloud/fog architectures for 5g services," in *21st Opto-Electronics and Communications Conference (OECC) held jointly with 2016 International Conference on Photonics in Switching (PS), EEE*, 2016.

[98] J. F. Riera, J. Batall, J. Bonnet and et al., "TeNOR: Steps towards an orchestration platform for multi-pop NFV deployment," in *2016 IEEE NetSoft Conference and Workshops*

(NetSoft). IEEE, 2016.

[99] "SONATA project," 2020. [Online]. Available: https://www.sonata-nfv.eu/content/sonata-project .

[100] C. J. Bernardos and C. Quintana, "5G-TRANSFORMER Project Grant No. 761536 Second periodic report of the project," 5G-Transformer EU, Jan. 2020. [Online]. Available: http://5g-transformer.eu/wp-content/uploads/2019/11/D7.5_Second_periodic_report_of_the_project.pdf.

[101] R. Ferrus, N. Kuhn and et al., "Virtualised hybrid satellite-TerrestriAl systems for resilient and fLexible future networks (VITAL), system Architecture: Final Report," VITAL, Jun. 2016.

[102] W. Dapeng and et al., "User-centric edge sharing mechanism in software-defined ultra-dense networks," *IEEE Journal on Selected Areas in Communications,* vol. 38, no. 7, pp. 1531-1541, 2020.

[103] V. Millnert, E. Bini and J. Eker, "Cost minimization of network services with buffer and end-to-end deadline constraints," *ACM SIGBED Review,* vol. 14, no. 4, pp. 39-45, Jan. 2018.

[104] M. Di Natale and J. A. Stankovic, "Dynamic end-to-end guarantees in distributed real time systems," in *Proceedings of the 15-th IEEE Real-Time Systems Symposium, pp. 215–227*, Dec. 1994.

[105] S. Jiang, "A decoupled scheduling approach for distributed real-time embedded automotive systems," in *Proceedings of the 12th IEEE Real-Time and Embedded Technology and Applications Symposium, pp. 191–198*, 2006.

[106] M. Alrifai and T. Risse, "Combining global optimization with local selection for efficient QoS-aware service composition," in *Proceedings of the 18th international conference on World wide web, pp. 881-890, ACM*, Apr. 2009.

[107] A. Dvir, Y. Haddad and A. Zilberman, "Wireless controller placement problem," in *15th IEEE Annual Consumer Communications & Networking Conference (CCNC), IEEE*, 2018.

[108] T. H. Tan, C. Manman, S. Jun, L. Yang, A. Etienne, X. Yinxing and D. Jin Song, "Optimizing selection of competing services with probabilistic hierarchical refinement," in *IEEE/ACM 38th International Conference on Software Engineering (ICSE), pp. 85-95*, 2016.

[109] A. Sheoran, P. Sharma, S. Fahmy and V. Saxena, "Contain-ed: An nfv micro-service system for containing E2E latency," *ACM SIGCOMM Computer Communication Review,* vol. 47, no. 5, pp. 54-60, 2017.

[110] H. Yang, J. Zhang, Y. Zhao, J. Han, Y. Lin and Y. Lee, "SUDOI: software defined networking for ubiquitous data center optical interconnection," *IEEE Communications Magazine,* vol. 54, no. 2, pp. 86-95, 2016.

[111] L. Hardesty, "Linux Foundation combines 6 networking projects into 1," SDX Central, (0nline), Jan. 2018. [Online].

[112] J. M. S. Vilchez, I. G. B. Yahia and N. Crespi, "Self-healing mechanisms for software defined networks," in *8th International Conference on Autonomous Infrastructure, Management and Security (AIMS)*, 2014.

[113] R. Mijumbi, "On the energy efficiency prospects of network function virtualization," arXiv preprint arXiv:1512.00215, 2015.

[114] K. Bijon, R. Krishnan and R. Sandhu, "Virtual resource orchestration constraints in cloud infrastructure as a service," in *Proceedings of the 5th ACM Conference on Data and Application Security and Privacy, pp. 183–194*, 2015.

[115] W. Stallings, SNMP, SNMPv2, SNMPv3, and RMON 1 and 2, 3rd Ed., chap. 9 pp. 251-256, Addison-Wesley Longman Publishing Co., Inc., 1998.

ECOLE DOCTORALE

Titre : Nouvelle proposition d'architecture et fonctions conséquentes au sein d'un orchestrateur pour assurer l'autonomie et la dynamicité de la gestion des réseaux futurs

Mots clés : gestion de réseau, orchestration, dynamicité, autonomie, aide à la décision, et service sur demande

Les réseaux 5G et IoT face à une explosion de la demande, deviennent plus complexes et difficiles à gérer. L'automatisation est limitée et les interventions humaines génèrent des erreurs. Côté forums, l'orchestration, les contrôleurs SDN, et la virtualisation réseau introduisent une dynamicité partielle. Côté recherche, la personnalisation impose agilité et intelligence. Cependant, l'orchestration est monolithique. La dynamicité et l'autonomie relatives au « on-demand » ne sont pas assurées. Aussi, face à ces nouveaux défis, nos contributions tentent de répondre à ces besoins. Notre première proposition, architecturale et organisationnelle introduit une couche pour concevoir et gérer les réseaux virtuels.

Notre orchestration est distribuée sur 5 couches afin de garantir autonomie et performances. Notre seconde proposition, fonctionnelle, supportée par une simulation, adresse la dynamicité des services « on-demand ». Notre dernière proposition soutenue par une analyse numérique, est une fonction d'aide à la décision, basée sur le SLA de haut niveau afin d'améliorer le ratio de services allouables. Face à ces besoins d'autonomie, de dynamicité, et d'intelligence du nouvel écosystème, notre recherche vise le graal du « zero-touch ».

Title: New architecture and function to improve autonomy, dynamicity, and intelligence of future network management

Keywords: network management, orchestration, dynamicity, autonomy, decision-making, and on-demand service

Abstract: 5G and IoT networks face an explosion in demand, and therefore become more complex and difficult to manage. Automation is limited and human intervention generates errors. On the standardization side, orchestration, SDN controllers, and network virtualization introduce partial dynamicity. On the research side, user centric services require agility and intelligence. However, the orchestration is monolithic. The dynamicity and autonomy relating to "on-demand" are not guaranteed. Thus, after 20 years of experience in the telecom industry, our contributions attempt to meet these new challenges.

Our first architectural and organizational proposal introduces a new layer to design and manage virtual services.
Our orchestration is distributed over 5 layers to guarantee autonomy and performance. Our second proposal, functional, supported by a simulation, addresses the dynamicity of "on-demand" services. Our last proposal supported by a numerical analysis, is a decision-making function, based on the high-level SLA in order to improve the ratio of allowable services. Faced with these needs of autonomy, dynamicity, and intelligence of the new ecosystem, our research aims at the holy grail of "zero-touch".

Institut Polytechnique de Paris
91120 Palaiseau, France