Multimedia Tools and Applications (2019) 78:24707–24737
https://doi.org/10.1007/s11042-019-7283-3

Dynamic network slicing management of multimedia scenarios for future remote healthcare

Alberto Huertas Celdrán[1] · Manuel Gil Pérez[2] · Félix J. García Clemente[3] · Fabrizio Ippoliti[4] · Gregorio Martínez Pérez[2]

Received: 19 September 2018 / Revised: 17 January 2019 / Accepted: 27 January 2019 /
Published online: 7 February 2019

Abstract

ICT solutions must meet the requirements demanded by challenging and complex scenarios such as remote care, which can be viewed as a combination of heterogeneous services using multimedia and home-care tools. Network Slicing emerged to this end, a paradigm tailoring the needs of any scenario whose specifications need to be met at all times. For its implementation, the network flexibility and resource control features provided by the Software-Defined Networking (SDN) and Network Functions Virtualization (NFV) techniques can allow the Network Slicing paradigm to manage the peculiarities established by any given scenario, taking a special consideration to multimedia scenarios with particular needs such as low latency and high bandwidth. However, existing Network Slicing approaches lack management mechanisms to understand *when* resources and services have to be changed or reconfigured to continue meeting the requirements, *what* elements would be involved in this updating process, and *how* changes would have to be performed. This article addresses this challenge by proposing an architecture able to manage the complete life cycle of Network Slices and to determine when, what, and how to dynamically orchestrate the resources and services so as to meet the scenario requirements. This SDN/NFV-enabled architecture allows managing the underlying infrastructure at run-time through policies, which use a formal Network Slicing information model based on ontologies also proposed in this article. Also, a complete use case is exercised to face the specific requirements of a given eHealth scenario with multimedia services, whose feasibility is demonstrated through a number of conducted experiments.

Keywords Network slicing · SDN/NFV techniques · Dynamic network management · Multimedia services

✉ Alberto Huertas Celdrán
ahuertas@tssg.org

Extended author information available on the last page of the article.

1 Introduction

Information and Communication Technologies (ICT) undergoes a radical change in recent years due to the arisen of complex scenarios demanding the fulfillment of a great number of heterogeneous and dynamic requirements. For example, the provision of certain challenging services in remote care scenarios, such as multimedia and home-care, requires real-time, flexible and automatic management systems able to control the scenario infrastructure and its communications. Coupled with the introduction of these new requirements, ICT has been strengthened with new and innovative technologies such as Software-Defined Networking (SDN) and Network Functions Virtualization (NFV). SDN is an approach to programmable networks built to separate the control and data planes, while NFV decouples the software implementation of Network Functions (NF) from the underlying hardware. The combination of both technologies entails a recognized number of advantages for network operators and service providers, as identified in [4]. In case of network operators, they can be strengthened in dynamism with the management of network resources and gain more flexibility in provisioning services, while in case of service providers they can achieve a significant improvement when administrating their infrastructure and provisioned services in real time. These benefits for both stakeholders come from a fully flexible and efficient control of their own assets, in addition to the deployment of virtual network infrastructure and services that can provide better availability, scalability, and efficiency.

Despite the aforementioned benefits of the SDN/NFV technologies, network management is a hard task because they do not provide an easy and formal way of meeting the specific requirements that particular challenging scenarios, like healthcare and multimedia, are demanding for their proper operation. To this end, the Network Slicing paradigm [27] emerged to manage the vast heterogeneity of services as well as the diversity and dynamism of the scenario requirements. The Network Slicing concept was initially introduced in 2010 referring to the possibility to combine network resources to enable their parallel use when sharing the same underlying infrastructure, as presented in [41]. Subsequently, a language semantics was defined in [10] with the goal of programming isolated Network Slices. Nowadays, the Network Slice concept may be defined as a "specific set of logical resources capable of providing certain services for diverse scenarios that demand different requirements".

As an example, Fig. 1 shows how current proposals manage a Network Slicing environment, which is comprised by two scenarios with different requirements that should be met by their given Network Slices: one *eHealth slice* for a healthcare scenario and another *Ultra HD Video slice* supporting a multimedia scenario. Each of them may be demanding different requirements, such as *high bandwidth* for the multimedia services or *high availability* for the healthcare ones, among others.

As shown in Fig. 1, the Service Provider checks out the Network Slice Blueprint Catalog to meet the established requirements by each scenario, in which a given *Network Slice Blueprint* is already onboarded with the description of the structure, configuration, and workflows to instantiate and control the Network Slices during their life cycle. As defined in [27], a Network Slice Blueprint is a *template* that describes how a Network Slice instance is to be created, which is defined at design or configuration time. Once the catalog is checked, the Network Slices that better fit the current scenario requirements are deployed. For each Network Slice, the Virtual Network Function (VNF) Manager uses a specific set of VNFs that are running on one or more Virtual Machines (VMs), sharing the same hardware networking infrastructure. Beyond network-centric functions implemented as VNFs, such as firewalls and access points, among others, the implementation of others types of virtualized

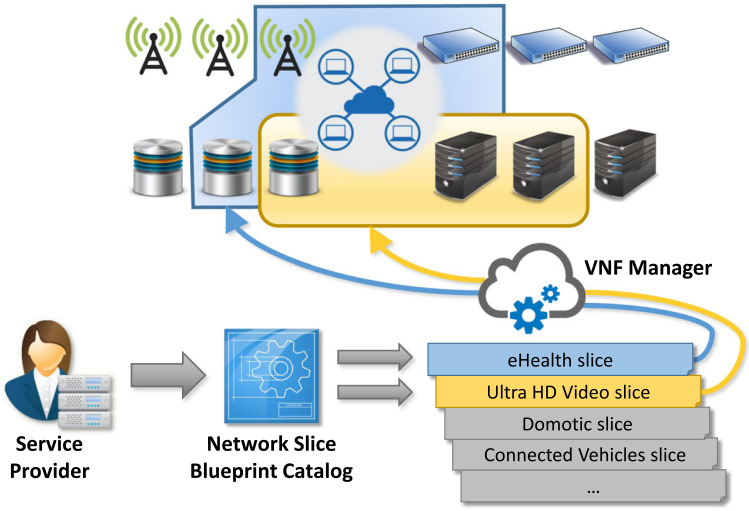

Fig. 1 Example scenarios implementing network Slicing with a common infrastructure

services can also be taken into account for the deployment of certain functions, which we can refer to as Virtualized Functions (VxF) as defined by [9]. It is worth noting that other Network Slices, as shown in Fig. 1, could be onboarded within the Network Slice Blueprint Catalog to manage other different scenarios with their own requirements, but all of them making use of the same underlying infrastructure.

Network Slicing is a new paradigm with high interest of actual study, which is mainly prompted by the incoming fifth generation (5G) mobile technology. New 5G scenarios are emerging in which SDN and NFV technologies have a fundamental role, both being the key driver to implement Network Slices. Recent research has focused on Network Slicing considering 5G networks, as the one proposed in [39], as well as the proposal of [30] for integrating the SDN architectures to enable Network Slicing. However, and despite the progress made by solutions such as the above, new mechanisms are required to dynamically manage the Network Slices and their resources that consider the specific requirements according to the target scenarios.

To face the previous challenges, this article substantially extends the solution presented in [18] by adding a new significant number of innovative contributions. In that contribution, a mobility-aware architecture implementing SDN/NFV techniques was presented by using a first version of the Network Slicing information model, but without exercising the architecture to get performance results and evaluate its feasibility. Due to this, the current article extends the another with the following contributions:

1. A revised formal definition of the Network Slicing information model in which the set of requirements demanded by a given scenario is implicitly added to the Network Slice Blueprint, and also adding a new Sub-network Blueprint to create sub-network instances forming the NFs that will run on physical/logical resources. The proposed Network Slicing information model has been developed and implemented using Semantic Web techniques, which can be publicly found at [48].
2. A new realistic use case with specific requirements of an eHealth scenario with multimedia services, in order to provide remote care assistance to show how our solution

addresses certain concerns to continue meeting those requirements. Multimedia services and their specific requirements will have important implications on such use case to demonstrate how using certain multimedia tools can be determinant in future remote healthcare environments.

3. A policy-based system approach that uses management policies to control the Network Slices life cycle dynamically, in which two types of policies are defined, intra-slice and inter-slice policies. The former are used to manage the elements making up the Network Slice itself, while the latter are responsible for switching from a given Network Slice to a different one.
4. A given number of experiments carried out to measure the time required to make decisions when applying intra- and inter-slice policies, as well as the time to enforce Network Slices meeting the established requirements. These experimental results will allow validating the feasibility of the proposed architecture, with the support of the intra- and inter-slice policies.

The remainder of the article is structured as follows. Section 2 discusses the main related work on Network Slicing and solutions managing Network Slices in multimedia scenarios. Section 3 presents a formal information model of Network Slicing based on ontologies. Section 4 shows a realistic use case in an eHealth scenario with a number of concerns, considering the specific requirements when provisioning multimedia services, which are addressed by our solution through the two types of policies presented in Section 5. Section 6 outlines the components of the proposed architecture to manage the Network Slices and their elements, whose feasibility is validated by the conducted experiments presented in Section 7. Finally, conclusions are drawn and future work suggested in Section 8.

2 Related work

5G networks need to integrate network services with varying performance requirements (e.g. high throughput, low latency, high reliability, high mobility, and high security) into a single physical network infrastructure. List of requirements that have been identified, as emphasized in [2], for the proper operation of multimedia services. 5G networks also need to provide each service with customized logical networks, such as the ones needed to transmit multimedia communication signals. In this regard, Network Slicing has recently been proposed as one of the key technologies to achieve the aforementioned goals. As any new technology, a number of research questions and open challenges have emerged last years associated to Network Slicing, as identified in [8]. As an example, the authors of [1] overview principal concepts as well as current standardization efforts in Network Slicing, identifying a number of open challenges and providing recommendations toward potential solutions.

Different scenarios and use cases were proposed in [23] with the goal of showing how Network Slicing can improve the current challenges and requirements. The authors brought three scenarios out: a *Smart City scenario* to manage the variety of different networks considered to acquire information, where IoT networks are run by different departments with distinct access privileges; a *Vehicle-To-Everything (V2X) scenario* to provide precise knowledge of the traffic situation to optimize traffic, reduce congestion, minimize accidents, and so on, not forgetting the highly reliable communication required; and an *E-health scenario* to enable life-critical services that monitor patient's vital signs (e.g. heart rate, pulse, blood pressure, etc.) in a reliable, fast, and secure way. The authors emphasize with these use

cases the need to manage enhanced broadband voice, data, and video communication services, such as real time interactive multimedia and high bandwidth video feed. Another work describing potential Network Slicing use cases is the one presented in [1]. Among such cases, it was presented a way of enhancing broadband access in densely populated areas (e.g. stadiums or open-air festivals) when enabling multimedia services such as ultra high definition video streaming. IP Multimedia Subsystem (IMS) –with services such as voice and interactive video streaming– was introduced in this work as a sub-network instance with which to deploy a given Network Slice with its multimedia services.

Some key requirements for Network Slicing are defined in [34]. Among the most relevant ones, the authors highlight the *network slicing resource specification* to manage the description of resources and NFs; *guaranteed slice performance and isolation* to enable the safe, secure, performance guaranteed service for multi-tenancy on common physical networks; *cross-network segment and cross-domain negotiation*, so that each segment of the network may be further divided into different administrative domains; *network slicing domain-abstraction* to be aware and independent of the domain to which they belong; *slice identification*, associated to privacy and security concerns of Network Slicing; and *OAM operations with customized granularity*, since different Network Slices' users (operators, customers) will have different requirements.

The authors of [33] set out Network Slicing as a combination of the SDN architecture with some essential capabilities supplied by NFV. In addition, they present an example scenario that combines SDN/NFV technologies to address the implementation of Network Slices, and thus approach response times and amount of bandwidth when dealing with multimedia services. Another solution is presented in [47], where the authors combine emerging technologies such as NFV, SDN, and cloud and edge computing to provide and manage Network Slices. They also underline the need to support a large number of slices and their deployment depending on subscribers' location. Concerning NFV-based solutions, the authors of [36] identified IMS as one of the most complex NFV instances, due to extremely low end-to-end latency and a high system availability. Moreover, a weak fault tolerance is another problem associated with IMS, which is addressed by the authors executing certain failover procedures after isolating faulty modules.

The authors of [50] showcase a 5G network slicing architecture together with a scheme for managing mobility between different access networks, as users move around while consuming multimedia services. Additionally, they also propose a joint power and sub-channel allocation scheme in spectrum-sharing two-tier systems based on Network Slicing. Another architecture for next-generation networks is designed and implemented in [24], which lies on NF decomposition and Network Slicing. Specific NF blocks for forming Network Slices and their roles and requirements are defined in the proposed architecture. Finally, the authors also analyze the impact of the architecture on multiple tenants. The concept of Network Slice as a Service (NSaaS) was introduced in [51] with the goal of helping operators to offer customized end-to-end cellular network as a service. This work also illustrates how operators can configure and manage NSaaS and APIs for customers. Another type of scenario where operators are interested is in delivering Information-Centric Networking (ICN) in 5G mobile networks by utilizing Network Slicing, as presented in [35], and how Mobility as a Service (MaaS) can be performed as a 5G-ICN Network Slice to meet MaaS objectives. This work also showed a videoconferencing slice in which users could request audio/video/text content from other third parties.

Network Slicing may impact several aspects of the 5G Radio Access Network (RAN) such as the protocol architecture, the design of NFs, or the management framework, as noted in [3]. This will have a direct implication when defining services for its commissioning, for

any type of application such as those based on multimedia services on which this article is focused. Regarding the efficient management of the elements making up the 5G RAN, a novel architecture is proposed in [7]. This architecture supports RAN slices according to the definition of a set of configuration descriptors that are able to characterize the features, policies, and resources of radio protocol layers. The work described in [19] presents an energy-aware and policy-based system oriented to the SDN/NFV paradigms, which allow the management of the network infrastructure at run-time through policies in a dynamic way. The authors of [20] propose a given admission control mechanism able to allocate network resources to different slices with the goal of improving the users' satisfaction while ensuring the slices' requirements. The authors performed several experiments that showed a higher user experience in individual slices and a higher scalability with large amounts of users. Also noteworthy is the work presented in [37], describing a resource allocation mechanism based on airtime assignment to achieve infrastructure sharing and slicing in Wi-Fi Access Points. This has the potential to be straightforwardly used in scenarios of wireless access infrastructure sharing, this being a conventional setting in many user-oriented scenarios.

Finally, it should also be pointed out that other works are providing solutions to exercise multimedia services over 5G RAN in critical scenarios. In particular, Public Safety is one of the most attention it is getting. A recent work, presented in [43], describes a public safety use case in which mission-critical services (e.g. video and data) are being used, in order to cope with the number of first responders consuming mission-critical video transmissions. The proposed use case entails the real-time management of large volumes of information and network traffic, especially in non-urban geographic areas where ubiquitous access is required for the tasks first responders need to perform.

Despite the progress made by the previous solutions, a substantial body of work is still required in the definition and management of Network Slices able to meet the requirements demanded by the scenario concerned. We propose in this article an architecture with innovative components that is in charge of managing and orchestrating the Network Slices, as well as policies to provide a flexible, efficient, and self-healing solution that complies with the specific requirements of multimedia scenarios, with services requiring real-time audiovisual communication.

3 Network slicing formal model

This section presents three ontologies that extend the current notion of Network Slicing. These are illustrated in Fig. 2, based on the common taxonomy proposed in [27] and the suggestions on network instantiation presented in [38]. To the best of our knowledge, the ontology-based formal models presented below are the first ones proposed in the literature, and they are based on the concepts and components found therein. We decided to model three different, but complementary ontologies rather than a single one for reuse reasons, where the three main concepts to define Network Slices are clearly differentiated by design.

The ontologies making up the Network Slicing formal model presented below can be found published at [48].

3.1 Network slice blueprint ontology

This ontology, shown at the bottom left of Fig. 2, focuses on the definition of the different components making up our Network Slicing concept, which are oriented to ensure the

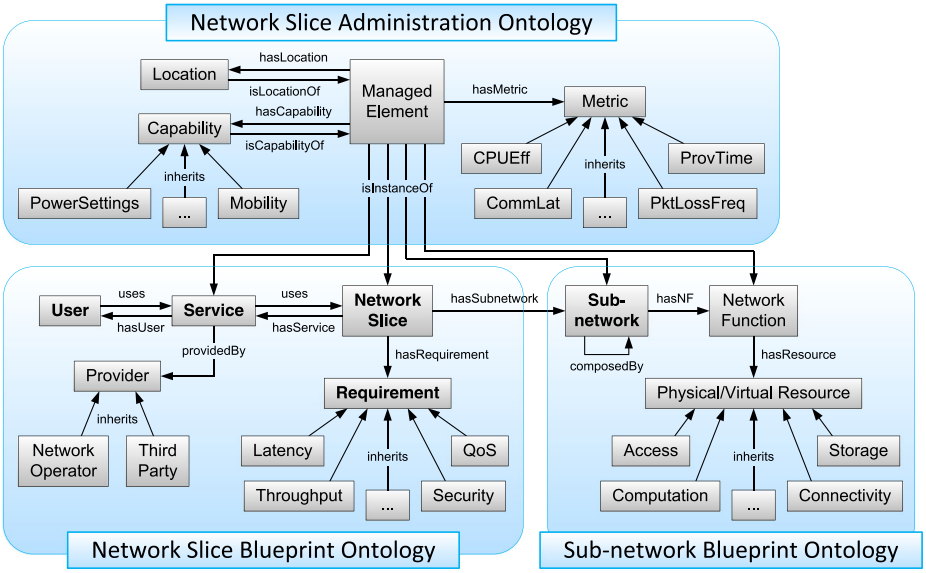

Fig. 2 The network slicing ontologies

provisioning of heterogeneous services. The main class of this ontology is *NetworkSlice* that represents the instantiation of every element contained by the *Network Slice Blueprint Ontology*. A given Network Slice has specific requirements, represented by the *Requirement* class, which are accomplished by one or more *Sub-networks* being responsible for the provisioning of *Services*. They are maintained by service providers, represented by the *Provider* class. The *Service* class is relevant because links all the concepts that describe the services being offered to users. As will be introduced in Section 4 with a motivation example, the Service class can be used to define and implement multimedia services with which to enable real-time audiovisual communications.

Finally, Providers could be a *Network Operator* or a *Third Party*, although this class could be extended to other types of providers, while the *User* class represents people consuming the service (or services) ensured by the Network Slice.

3.2 Sub-network blueprint ontology

The *Sub-network Blueprint Ontology*, illustrated at the bottom right of Fig. 2, is able to create sub-networks to form one or more Network Slices. The *Sub-network* class is the main concept of this ontology and it can be comprised by other sub-networks. A Sub-network has one or more NFs running on top of *Physical/Virtual Resources*, which will provide the necessary support for these NFs to function properly on the network. The *NetworkFunction* class includes every kind of resource that can be used or deployed within a given IT infrastructure, which can be related to computation, storage, or networking, for example.

A given Sub-network represents a way of reusing a combination of fundamental NFs that provide similar features shared by different Network Slices. The combination of all this information represents a Sub-network Blueprint, which is a type of template of how to deploy a Sub-network meeting Network Slice requirements.

3.3 Network slice administration ontology

As shown at the top of Fig. 2, this ontology aims to manage the dynamism and mobility of the critical components represented by the *NetworkSlice*, *Service*, *Sub-network*, and *NetworkFunction* classes defined in the two previous ontologies. All of them are instances of the *ManagedElement* main class. It is worth noting that all these elements included in the three ontologies have been modeled by following the Common Information Model (CIM) language, formally defined by the DMTF as a standard in [5]. CIM provides a common definition of management information for systems, networks, applications, and services. As a consequence, all these elements are defined as instances of the *ManagedElement* class, which is the main element for a given model in CIM.

The *ManagedElement* class identifies a generic entity that has some features, represented by the *Metric*, *Location*, and *Capability* classes. The *Metric* class refers to any indicator useful in order to measure the performance of the Network Slices and their elements. In the proposed use case, as discussed below, this indicator refers mainly to Quality of Service (QoS) values if the managed entity is a service. As a proof of concept, we defined four different metrics: *CPUEff* indicating the percentage of CPU used by the managed elements; *CommLat* denoting the latency experienced by the communications; *PktLossFreq* (packet loss frequency) pointing out the availability and QoS of the Network Slices communications; and *ProvTime* reflecting the time required to provide a given service or slice. These classes represent a pool of potential metrics that could be extended with additional ones, which we consider to measure the status of the Network Slices and their elements. It is worth noting that these metrics do not pose aspects provided by our policy-based system, although they could be aspects that the latter could also have.

Finally, the *Location* class shows information about the relative position within the environment concerned. This is the top-level class of a hierarchical model for location, having five predefined subclasses; namely (from low to high accuracy): *Continent*, *Country*, *Region*, *City*, and *Position*, when the latter can in turn establish the *Geographical* or *Absolute Position* of any element. Note that they are not shown in Fig. 2 to ease its understanding. In addition, as depicted in the figure, the *Capability* class includes other subclasses such as *PowerSettings* and *Mobility*, among others. While the former enables the optimization of energy efficiency, the latter has a key role when services are provided in highly dynamic environments.

4 A motivating example: Multimedia in remote care

This section shows a use case that highlights the necessity of managing Network Slices that consider different requirements of dynamic healthcare environments. One of the main pillars of the healthcare of the future is remote care. Remote care refers to the idea of removing the hospital borders and allowing people to stay in their own homes, providing person-centered medical services and technologies to support their individual care. In this context, multimedia systems play a key role by enabling a direct and real-time audiovisual communication between the hospital staff and patients.

In this remote care scenario, unexpected clinical events might happen at given times provoking changes in the requirements of the remote clinical environment. Among potential situations, we highlight unexpected health events characterized by the variation of certain vital signs of patients located at home. This scenario consists of three phases that are

presented below: a) the initial scenario, b) when the unexpected clinical event occurs, and c) after the mitigation takes place.

4.1 Initial situation: Sensing vital signs in remote care

The proposed clinical scenario defines a Network Slice (NS), named as *RemoteCare NS*, which is focused on providing and managing specific elements required to sense different vital signs of patients located at home (e.g. medical devices, services, and communications). It is worth noting that the RemoteCare NS will have several patients sharing the NS services and resources.

Despite security and privacy concerns are outside the scope of this work, it is important to highlight that having different patients sharing the NS resources affects to the intra-slice data security and privacy. In this sense, some mechanisms and protocols at different levels (users, services, and resources, as indicated in [40]) are required to protect the users' privacy and security. Regarding the inter-slice security, in [22] it is emphasized that isolation among Network Slices can be achieved via data plane isolation and control plane isolation, which can be protected by using cryptographic primitives. For example, a hybrid approach is presented in [25] to secure the healthcare data transferred between individual patients and medical staff, fully protected from any other type of communication, by using linear network coding and re-encryption based on ElGamal cryptography. More advanced cryptographic mechanisms are currently being considered, such as homomorphic encryption. In this context, a system and method for homomorphic encryption environment is proposed in [44] for a healthcare network, where queries result in encrypted data without data servers know its content, and how to infer it.

Figure 3 shows the following elements considered by the RemoteCare NS:

- Requirements:
  - high availability, security, privacy, and resiliency.
- Resources:
  - medical devices located at the patients' home such as heart monitors, pacemakers, defibrillator, infusion pumps, and wearables. This equipment is able to sense medical data from patients and actuate directly over them;

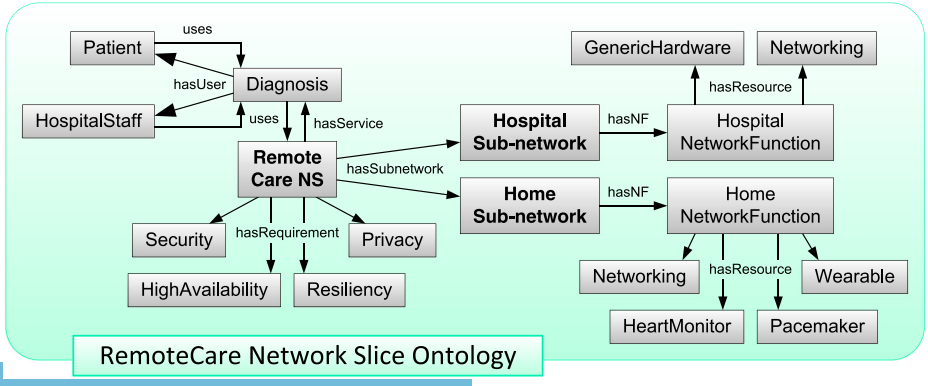

Fig. 3 Main elements of the RemoteCare NS

  - generic hardware equipment located at the hospital side to host the clinical services; and
  - networking equipment located both at home and in hospital to enable the communication between both sides.

- Services:
  - diagnosis services located at the hospital in charge of collecting and analyzing the medical data sent over secure channels, having such services specific security and privacy methods to protect the sensitive information.

- Users:
  - hospital staff consuming diagnosis services to make decisions about medical treatments; and
  - patients receiving remote care.

How to supply diagnosis services with protection mechanisms to ensure security can be achieved by cryptographic methods, as described above, while privacy of users' data has also been widely studied, as in [17] where a privacy-preserving and context-aware solution based on policies is proposed.

At this given point, the provisioning of the diagnosis services afforded by the RemoteCare NS is working as expected. Furthermore, the resources are not overloaded and the hospital staff can access the diagnosis services without experiencing problems. However, the main issue in this point (***Concern 1***) would be that unexpected and massive clinical events, such as a flu epidemic, would cause a worsening of the evaluation metrics of the diagnosis services when the number of patients increases massively. In this case, a mechanism able to administrate Network Slices should trigger a dynamic adjustment of the RemoteCare NS resources by taking into account the location and mobility of users, the Network Slices in place, and their resources. These adjustments can vary from adding new (or more) computational resources to the implementation of new hardware technologies that allow complying with the requirements established by the Network Slice.

4.2 Unexpected clinical event: Multimedia in remote care

The second phase of this use case shows the occurrence of an unpredictable event, e.g. the deterioration of some vital sign of the patients located at home. Nowadays, different types of patients require continuous control of their vital signs. Examples of that could be patients with chronic diseases who receive care at home, patients who recently undergone a surgery and are being treated at home, or elderly with mobility problems being monitored to avoid falls or clinical issues.

The unpredictable event is detected thanks to the monitoring and diagnosis processes performed by the elements belonging to the RemoteCare NS, introduced in the previous section. However, in this point arises the ***Concern 2***, which highlights the necessity of establishing two different types of multimedia services to monitor patients. We consider the establishment of i) *video conferences* between the medical staff and the patients to notify them the problem as well as provide further information to the medical staff; and ii) *night video supervision* to detect strange behaviors automatically, such as falls or no movements for a long period of time. Since the requirements and resources of the new multimedia

services are quite different to the RemoreCare NS ones, it is required the management of an additional Network Slice (NS), named *Multimedia NS*. It is worth mentioning that the Multimedia NS will be working on parallel to the RemoteCare NS.

Figure 4 shows the following elements considered by the Multimedia NS:

- Requirements:
  - low latency and high bandwidth.
- Resources:
  - necessary multimedia equipment located at the patients' home such as video cameras, surveillance devices, and audio equipment. This kind of equipment is critical to establish audiovisual communications with the patients in order to detect problems in real time;
  - generic hardware equipment and videoconferencing components located at the hospital side to host multimedia services; and
  - a 5G network infrastructure (located everywhere) to enable the communication between both sides ensuring low latency and high bandwidth.
- Services:
  - telecare and multimedia services such as High Definition (HD) or 4K video streaming to enable real-time communications between the patients and the medical staff. In addition, we also consider a night video monitoring service to detect falls or strange behaviors of patients automatically.
- Users:
  - hospital staff consuming multimedia services to make decisions about the medical treatments; and
  - patients receiving remote care.

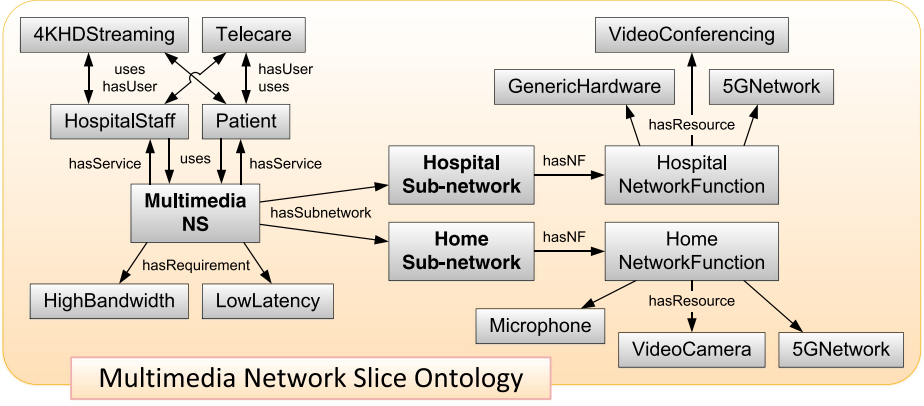

Fig. 4 Main elements of the multimedia NS

4.3 Post-mitigation scenario: Dismantling multimedia

Finally, the last phase of this use case consists in dismantling the *Multimedia NS* to release resources and ensure their optimal usage when patients' vital signs are between the normal range of values. In this context, the main issue (***Concern 3***) would be to have an automatic mechanism capable of detecting when the context or situation changes and when a given slice is not required anymore.

5 Network slices management policies

This section presents a policy-based approach to define management policies so as to control the Network Slices and their resources dynamically. These policies are translated into a set of rules to manage the resources state or replace a given Network Slice by instantiating a different one. They are categorized into two different groups, *intra-slice* and *inter-slice* policies, managing the Network Slice life cycle dynamically but with a sharp difference. While the intra-slice policies are used to manage the elements that compose the slice itself, e.g. adding a new resource to continue meeting the established Network Slice requirements, the inter-slice policies are responsible for switching from a given slice to a different one, e.g. due to the changing of some requirements of the services being provided. In both cases, they are defined according to the ontologies proposed in Section 3.

To manage the Network Slices and their resources, our two types of policies take into account diverse metrics oriented to measure different aspects and resources provided by the Network Slices. On the one hand, generic metrics such as the ones proposed in [21] are oriented to ensure the QoS, security, or availability of the resources belonging to different contexts. On the other hand, context-aware metrics are focused on controlling aspects related to the environment or context in which Network Slices are deployed. Following the remote care use case presented in Section 4, context-aware metrics could measure the number of frames per second in multimedia services or the heart rate in healthcare, among others. Additionally, it is important to note that policies consider the geographical location of NSs and their resources. In this context, combining the previous metrics with the location of physical and virtual resources our solution is able to manage and address the mobility issue identified in Section 1.

Table 1 Examples of suitable metrics for defining Network Slices management policies

Metric	Abbreviation	Scope
Communication Latency	CommLat	General (Communication)
Packet Loss Frequency	PktLossFreq	General (Communication)
Provision Time (s)	ProvTime	General (Service)
CPU Efficiency	CPUEff	General (Hardware)
Frames per second (s)	AcqTime	Contextual (Multimedia)
Jitter	Jitter	Contextual (Multimedia)
Image resolution	ImgRes	Contextual (Multimedia)
Percentage of blood oxygen	SpO2	Contextual (RemoteCare)
Heart rate	HeartRate	Contextual (RemoteCare)
Temperature	Temp	Contextual (RemoteCare)

Table 1 shows some examples of evaluation metrics that can be used to define policies. The first four ones have been used in the ontology of Fig. 2, where their class names are represented with abbreviations. In the following two subsections, several examples of policies will be introduced to address the potential use case discussed in Section 4. It is important to notice that we propose a generic policy-based solution able to manage specific policies aware to given scenarios.

5.1 Intra-slice policies

The policies in this category, addressing ***Concern 1***, evaluate different metrics of the resources provided by a given Network Slice to eventually perform some healing operations, according to which metric is raising some warning alerts. This could be applied in a situation as the one presented in Section 4.1 (initial scenario), where location values are relevant to provide that specific service efficiently.

As an example, the intra-slice policy shown below adds more computational resources (*#Cp*) to ensure the service (*#DiagnosisServ*) provided by a Network Slice (*#RemoteCareNS*) as a corrective measure to the increment of patients, due to a massive clinical event such as people infected by a flu epidemic.

Policy(#1, #IntraSlice):
NetworkSlice(#RemoteCareNS) *hasService* Service(#DiagnosisServ) ∧ #DiagnosisServ *hasMetric* Metric(#CPUEff *hasValue* #Orange) ∧ #DiagnosisServ *hasLocation* ?any *isLocationOf* Resource(#Cp) ⟶ *addResource*(#RemoteCareNS, #Cp)

This kind of policy considers the location of available computational resources (*#Cp*) to cover the necessities provoked by a flu epidemic and maintain the QoS of the diagnosis service. Alerts in this policy are identified with an *#Orange* threshold and refer to the metric measuring the CPU Instance Efficiency (*#CPUEff*), due to the CPU of the resource where the service becomes congested by the large amount of patient activity. Since service requirements are not changing, the Network Slice remains the same as before, but some adjustments are performed.

5.2 Inter-slice policies

The expected outcome of an inter-slice policy is to manage the activation or the deactivation of certain Network Slices according to the location of the services and resources. As an example of this kind of policy, two different inter-slice policies are defined below. Both could be applied in situations analogous to the concerned and managed scenarios of Sections 4.2 and 4.3, respectively.

The first inter-slice policy (ID *#2*) refers to a similar situation of the scenario shown in Section 4.2, which addresses ***Concern 2***.

Policy(#2, #InterSlice):
NetworkSlice(#RemoteCareNS) ∧ NetworkSlice(#MultimediaNS) ∧ #RemoteCareNS *hasMetric* Metric(#HeartRate *hasValue* #Red) ∧ #RemoteCareNS *hasMetric* Metric(#Temp *hasValue* #Red) ∧ #RemoteCareNS *hasLocation* ?any *isLocationOf* #MultimediaNS ⟶ enableSlice(#MultimediaNS)

This policy analyzes the overall status of the provided diagnosis service within the Network Slice that is always on (*#RemoteCareNS*). It shows how it is possible to dynamically activate a new Network Slice with different requirements so as to include new multimedia functionality to the remote care scenario. The corrective measures adopted by Policy #2 highlight the consequences after the policy has been triggered. When some critical medical problem is detected through the deterioration of the patient's vital signs, values go beyond a given threshold identified by a *#Red* alert, the policy will check the current location of the existing Network Slice. In this case, the metrics used by this policy are Heart Rate (*#HeartRate*) and body temperature (*#Temp*). The reaction will be to activate a Network Slice (*#MultimediaNS*), which is going to be deployed geographically in the same place as the RemoteCare NS. Additionally, our solution allows the medical staff to define customized policies for specific patients. In this context, a variant of this Policy #2 could be to have a higher or lower threshold of the heart rate and temperature metrics to deploy the Multimedia NS.

The second inter-slice policy shown below (ID *#3*) can be applied to address the ***Concern 3*** explained in Section 4.3.

$$\textbf{Policy(\#3, \#InterSlice)}:\ \text{NetworkSlice}(\#\text{RemoteCareNS}) \wedge \text{NetworkSlice}(\#\text{MultimediaNS}) \wedge \#\text{RemoteCareNS}\ \mathit{hasMetric}\ \text{Metric}(\#\text{HeartRate}\ \mathit{hasValue}\ \#\text{Green}) \wedge \#\text{RemoteCareNS}\ \mathit{hasMetric}\ \text{Metric}(\#\text{Temp}\ \mathit{hasValue}\ \#\text{Green}) \wedge \#\text{RemoteCareNS}\ \mathit{hasLocation}\ ?\text{any}\ \mathit{isLocationOf}\ \#\text{MultimediaNS} \longrightarrow \text{disableSlice}(\#\text{MultimediaNS})$$

This policy is able to restore a normal state by disabling the Multimedia NS used during the unexpected medical situation. Thanks to this policy it is possible to automatically detect the new overall conditions and disable the Multimedia Slice (*#MultimediaNS*) when the values of the patient's temperature and heart rate are comprised between the green thresholds.

As seen in this section, the intra- and inter-slice policies proposed in this article demonstrate how they are capable of managing Network Slices dynamically, taking into consideration mobility aspects of both users and network resources through a set of suitable metrics.

6 Architecture to manage network slicing

The automation of the provisioning of innovative services needs to rely on an agile architecture able to adapt to a highly dynamic context. Figure 5 shows the main elements of the proposed architecture, which has been developed starting from the ETSI reference model published in [6] whose components are represented on a blue background. Our contribution, which is shown in orange instead, manages the Network Slices and their resources dynamically with a policy-based approach.

In what follows, Section 6.1 describes the elements proposed by the ETSI architecture, while Section 6.2 introduces our vision in detail.

6.1 ETSI architecture

This section shows the elements that make up the proposed architecture in charge of managing the Network Slices given their mobility and particular requirements. Figure 5 shows

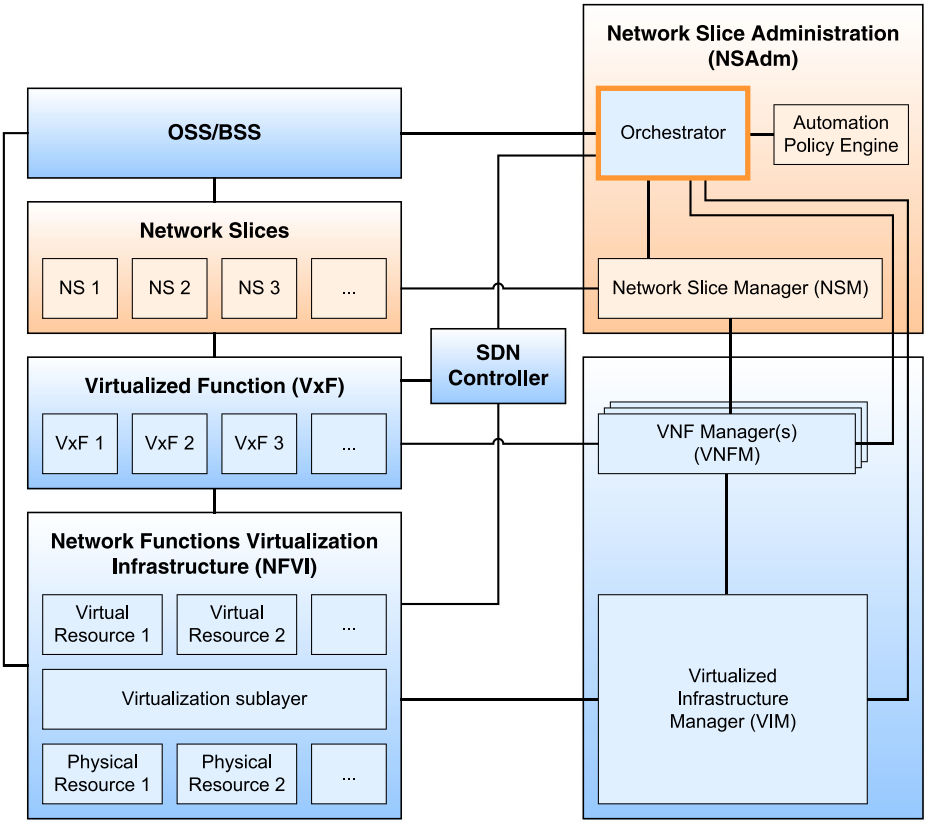

Fig. 5 Main elements of the proposed architecture

that the *Virtualized Infrastructure Manager* (VIM) is responsible for the *Network Functions Virtualization Infrastructure* (NFVI) management and the resource monitoring. The VIM is able to create, control, monitor, and dismantle the whole life cycle of VMs that are instantiated on generic *Physical Resources* or equipment making up the *Network Slices* through the *Virtualization Sublayer*. Physical and Virtual Resources can range from computation and storage elements to SDN-oriented network infrastructure (e.g. physical/virtual switches in addition to *SDN Controllers*) so as to decouple data and control planes. Although it has not been introduced into the figure for simplicity, a Wide Area Network Infrastructure Manager (WIM) could also be added to denote the wide area network counterpart of the VIM.

The *VNF Managers* (VNFM) are in charge of the configuration, management, and monitoring of each of the VxFs that make up the Network Slices, where a given VNFM is associated with managing a particular VxF. Each manager is able to deploy, dismantle, and configure its VxFs on-demand with the functionality provided by the Network Slices. Examples of VxFs can range from network functions such as load balancers or firewall, to general services like video streaming or voice. The VxFs run one or more VMs exposed and deployed in the NFVI. On the other hand, the SDN paradigm has the ability to decouple the *data plane*, where forwarding elements are located, from the *control plane*, where routing decisions are made. The main control element is called *SDN Controller*, which manages

multiple network elements belonging to the data plane. The SDN Controller allows the proposed architecture to control the network communications in real time between the elements of the Network Slices with the goal of ensuring aspects such as security, privacy, and QoS, among others.

Lastly, the *Orchestrator* automatizes the management of the aforementioned elements. This component communicates with the VIM, the VNFM(s), and the SDN Controller to schedule their tasks as well as to receive information about the state of the scenario, thus updating its catalogs. The Orchestrator also communicates internally with other components that are essential to manage the life cycle of the Network Slices, which are outlined below. Finally, the Orchestrator is linked to the *Operations Support System/Business Support System* (OSS/BSS) that serves the customer services.

6.2 Network slice administration (NSAdm)

The *Network Slice Administration* (NSAdm) includes a number of elements that directly deal with the life cycle of the Network Slices. These components, shown on an orange background in Fig. 5, represent a solution able to address the concerns about mobility, which have been discussed in the previous sections.

The *Automation Policy Engine* is the component in charge of making decisions about the management of the Network Slices. This manages the intra- or inter-actions taken into account the policies defined by the network administrator as well as the Network Slice context information and the location of the resources, which are considered by our architecture to control the Network Slices and their resources dynamically. These policies are categorized into two different groups, intra-slice and inter-slice policies, as presented in Section 5. The decisions made by the Automation Policy Engine are processed and automated by the Orchestrator, which we empower with innovative features. In fact, it becomes responsible for the automation of the processes related to the Network Slices management based on the outcomes chosen by the Automation Policy Engine. Specifically, once an intra- or inter-slice policy is triggered by the Automation Policy Engine the Orchestrator communicates with the *Network Slice Manager* (NSM) to perform the corrective measures to the given Network Slice. This manager handles the different Network Slices and Sub-networks and monitors their state. This information flows back to the Orchestrator in order to update the catalogs.

All the information considered by the *Orchestrator* and the *Automation Policy Engine* to perform the previous tasks is depicted in Fig. 6. The catalogs, resources, and instances managed by the ETSI model are shown on a blue background, while the catalogs, instances,

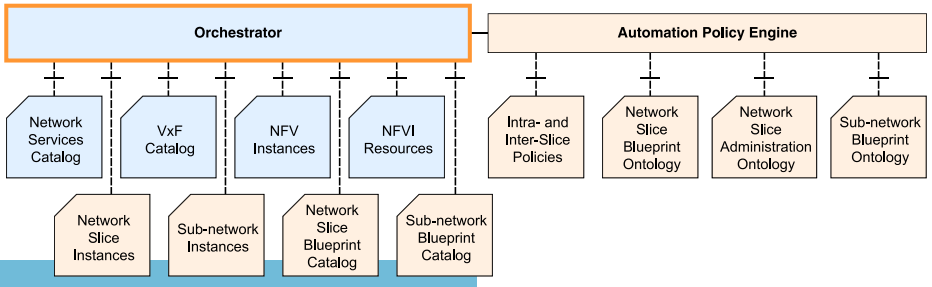

Fig. 6 Catalogs and databases of the proposed architecture

policies, and ontologies introduced by our proposal so as to manage the Network Slices and their resources are shown in orange.

6.3 Orchestration to manage Network Slices

This section showcases the number of steps performed by the Orchestrator of the proposed architecture to enable a new Network Slice. As Fig. 5 shows, once the Orchestrator receives new actions to be enforced it interacts with the different components belonging to the SDN, VxF, and VI layers for their enforcement.

With the motivating example of Section 4 in mind, Fig. 7 shows the different steps that are performed by the Orchestrator and the VI/VNF/NS/SDN managers to enable the new

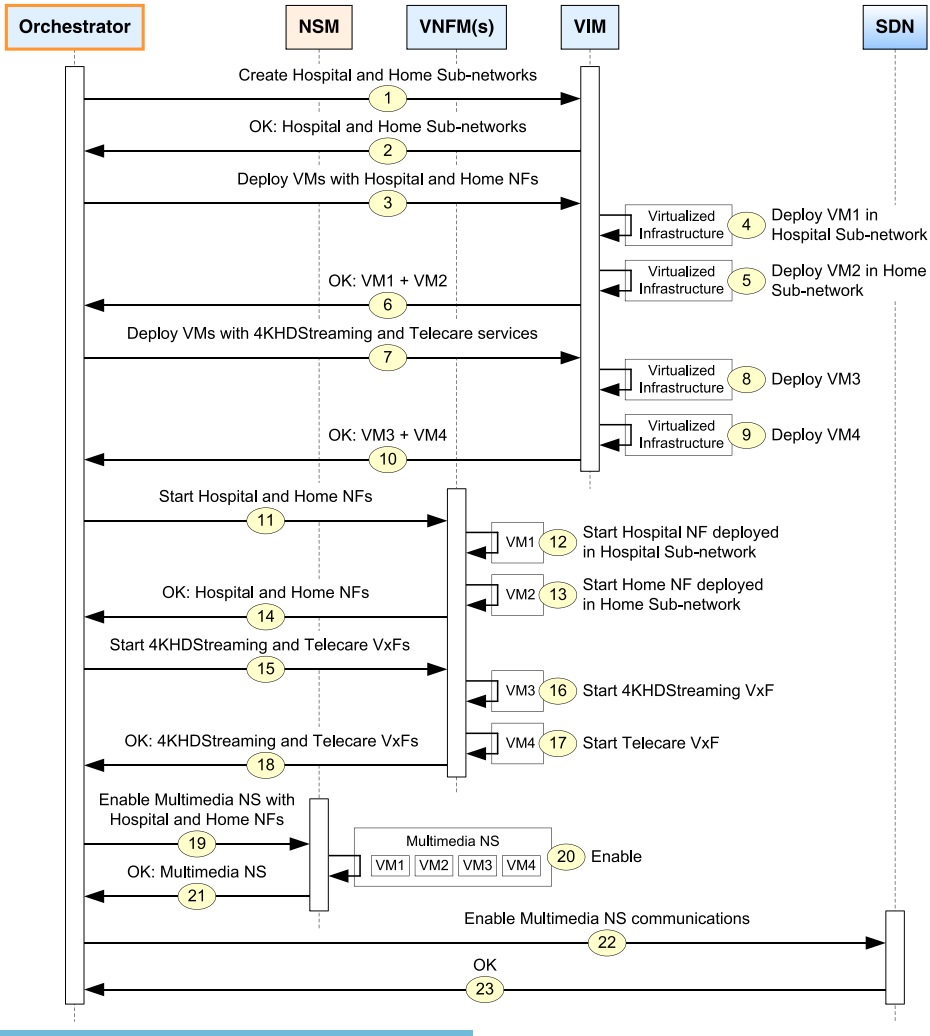

Fig. 7 Sequence diagram with the interactions between the Orchestrator and the SDN/NFV resources to enable a new Network Slice

Multimñedia Network Slice indicated by the Policy #2 of Section 5. To this end, the Orchestrator has the knowledge about the status of the whole infrastructure thanks to the different catalogs and instances maintained by itself. In this context, since the Multimedia NS is composed of two Sub-networks with a Network Function each one of them (Hospital and Home NFs) and two services (4KHDStreaming and Telecare), the Orchestrator firstly sends a request to the VIM asking for the creation of the Hospital and Home Sub-networks in the existing infrastructure (Step 1), and then checks in its own catalogs the status of the existing VMs. The infrastructure is optimized to the current running services, therefore, it is required to deploy two new VMs for the previous NFs and two additional for the services. For that, the Orchestrator contacts the VIM again to deploy two VMs with the Hospital and Home NFs (Step 3 in Fig. 7), which will be allocated in the two previous sub-networks. Once the VIM receives the request, it instantiates the *VM1* with the Hospital NF (Step 4), *VM2* with the Home NF (Step 5) and replies the Orchestrator with the status and details of both VMs (Step 6). The previous process is repeated to deploy the *4KHDStreaming* and *Telecare* services in *VM3* and *VM4* respectively (Steps from 7 to 10). Once the VMs have been deployed, the Orchestrator directly asks the associated VNFM to start the Hospital and Home NFs (Step 11), which will run in its corresponding sub-network. The VNFMs start both VMs with their NFs (Steps 12 and 13) and confirms the situation to the Orchestrator (Step 14). In this point, the previous process is repeated to start the 4KHDStreaming and Telecare services (Steps from 15 and 18). At this point, the resources required to enable the Multimedia NS are ready, so the next step is to notify the NSM about the situation (Step 19).

Once the NSM is aware, it associates and configures the VxFs (NFs and services) to the created Multimedia Network Slice (Step 20). In relation to Step 21, the NSM confirms the Orchestrator that the new Network Slice has been deployed, whose instance is composed of the available physical infrastructure, sub-networks, VMs, NFs, and services. Finally, the last step consists in communicating with the SDN Controller to enable the communications between the different NFs and VxFs belonging to the new Multimedia NS (Steps 22 and 23).

7 Experimental results

This section shows several experiments performed to measure the time required to make decisions about changing the Network Slices and their configurations, as well as enforcing the deployment of these Network Slices.

7.1 Deployment of the architectural components

We deployed the proposed architecture to demonstrate its proper functioning and measure its throughput and scalability. In this context, the representation of the information (ontologies and policies) and the decision-making process (Automation Policy Engine) are based on Semantic Web techniques. The Network Slice Blueprint Ontology, Sub-network Blueprint Ontology and Network Slice Administration Ontology, shown in Section 3, are formally defined and implemented in OWL 2 (Web Ontology Language), as specified in [26], being generated with the Protégé tool that can be found in [46]. The implementation of the previous ontologies in OWL 2 can be found and downloaded in [48]. We chose OWL 2 due to it was specifically designed as an ontology language, being an open standard and the main ontology language used nowadays in Semantic Web. On the other hand, the semantic

rules defining the policies of Section 5 are expressed in SWRL (Semantic Web Rule Language), as specified in [15]. SWRL includes a type of axiom, called *Horn clause logic*, of the form $if...then...$, being the most widely used solution in Semantic Web today.

Once the ontologies and policies have been defined in the proposed architecture, we show the process to translate automatically from them to a set of actions oriented to create or modify NSs. For that, a semantic reasoner, implemented by the Automation Policy Engine component of Fig. 5, infers new knowledge that decides whether the consequent of a given policy should be enforced. We used Pellet to this end, which can be found in [42], as semantic reasoner that receives ontological models generated by the Interpreter component (also implemented by the Automation Policy Engine component) and returns inferred models with new knowledge. The Interpreter generates ontological models with the information shaped in the ontologies and policies. Finally, the Automation Policy Engine component is in charge of translating the queries performed automatically by the architecture into SPARQL queries, formally defined in [49], which are applied to the inferred model to get the result.

Specifically, the Interpreter generates an ontological model from the Network Slice Blueprint ontology. This model is sent to the Reasoner to obtain the inferred model with new information inferred from the ontologies. Once the Interpreter receives the inferred model, it updates it with the new information provided by the Reasoner according to the policies defined by the administrator. When a query is performed, the Interpreter invokes the Engine with the latest inferred model and the query. Finally, the Engine applies the appropriate SPARQL queries to the inferred model and returns the result.

The rest of the architectural components of Fig. 5 were implemented with open source tools for SDN/NFV support within a fully virtualized network environment. Specifically, such tools were chosen to implement the components depicted in Fig. 5 on a blue background, in order to follow the ETSI reference model detailed in [6]. To implement the role of the VIM we used OpenStack (Liberty release), available at [32], to instantiate the different virtual resources such as the VMs in which the VxFs will run. We chose OpenStack since it is the de-facto open-source cloud management platform compliant with many tools for orchestration and management of the network control layer.

As Orchestrator we covered its role by using the Open Baton framework that is described in [29], which is an ETSI MANO compliant open source tool following the ETSI specification defined in [6]. In this case, we chose Open Baton (version 3.0.1) to orchestrate the elements belonging to the SDN/NFV planes, but also for providing a generic VNFM to manage the life cycle of the different VxFs in addition to support autoscaling and fault management features, understanding the Topology and Orchestration Specification for Cloud Applications (TOSCA) standard language specified in [28], etc. Finally, we made used of OpenDaylight (Beryllium release) as SDN Controller, available at [31], because this is able to control the different virtual switches deployed in the OpenStack infrastructure installing in their flow tables the OpenFlow rules received from the Orchestrator. We chose OpenDaylight because it is considered the industry's de facto SDN platform. Finally, it is worth noting that the previous elements have been deployed in virtual machines running on top of a server. In a nutshell, we have two levels of virtualization: the first one where the server virtualizes the management components (OpenStack, Open Baton, VNFM, OpenDaylight, and the Automation Policy Engine); and the second virtualization level, provided by OpenStack, where we deployed the scenario described and used in the next sections, which has several NS composed by different networks, routers, switches, VMs, and VxFs.

7.2 Decision-making process

We conducted several experiments with the aim of measuring the throughput and scalability of the decision-making process, which is implemented by the Automation Policy Engine of the proposed architecture (see Fig. 5). These experiments were intended to deal with two questions:

– Is the decision-making process time acceptable?
– How it scales with different intra- and inter-slice policies?

As experimental setting, the proposed framework and the conducted tests were carried out in a dedicated PC with an Intel Core i7-3770 3.40 GHz, 16 GB of RAM, and an Ubuntu 18.04 LTS as operating system. The results shown below have been obtained by executing the experiments 100 times and computing their arithmetic mean. To measure the performance of the decision-making process we performed several executions with different level of complexity. By increasing the number of individuals of the ontologies and the semantic rules will provoke an increment on the number of statements, and therefore on the complexity of executions. The number of individuals contained in our ontologies is referred as *population*. This was randomly generated, but in a controlled way to achieve a real distribution of the elements composing the environment. Table 2 depicts the number of elements used in our environment and their percentages.

Regarding the first question highlighted in this section, i.e. how acceptable the decision-making times are, we defined an initial population of 30,000 individuals that is increased with other 30,000 individuals in each new step. Table 3 shows the complexity of the proposed ontologies (relationships between the individuals and the statements generated by the semantic reasoner).

Figure 8 depicts the time, measured in milliseconds (ms), used by the semantic reasoner to validate the ontology considering different population groups (shown in Table 3). By evaluating the trade off between the increase of individuals and the time required by the decision-making process, we can see that the proposed solution can support a very large number of individuals within a reasonable time. Furthermore, the linearity property behind these results allows us to infer that a better computer system setting would obtain lower reasoning times.

Also related to the previous experimentation, we performed an additional test with the goal of checking how policies affect to the scalability of the proposed solution. This test will answer the second question highlighted in this section. To this end, we established several percentages of policies related with the persons contained in our population groups. Fig. 9 shows the variation of the time required by the decision-making process when the

Table 2 Individual distribution of population

Element	Individuals	Percentage	Element	Individuals	Percentage
Network Slice	1	0.9%	Resource	15	13.0%
Sub-network	4	3.5%	Hospital Staff	4	3.5%
Requirement	4	3.5%	Patient	50	43.5%
Service	5	4.3%	Other	20	17.4%
Network Function	12	10.4%	Total	115	100%

Table 3 Number of individuals and statements per population

Population	0	1	2	3	4
Individuals	30,000	60,000	90,000	120,000	150,000
Statements	326,543	645,943	979,342	1,300,291	1,645,352

population (see Table 3) and the number of policies change. As observed, policies have a very low impact in our framework, around a few milliseconds.

As main conclusion of this section, we have demonstrated with the previous experiments that when the number of individuals/statements is linearly increased in our ontology, the decision-making process time also increases linearly. Further, the semantic rules that form the policies do not have an important impact on the decision-making process time.

7.3 Reaction and virtualized enforcement

Once the Automation Policy Engine decides to enforce given actions, either intra-slice actions (e.g. to adjust the current Network Slice to continue meeting the requirements) or inter-slice actions (e.g. to deploy a new Network Slice), the Orchestrator and the NSM come into play to deploy and enforce such actions. In this context, we conducted several experiments so as to evaluate the feasibility of the deployment blocks of our architecture, by measuring the performance times of the following two cases to start up:

– The RemoteCare NS introduced in Section 4.1, through the deployment of three VMs: two with the Hospital and Home NFs and the other to implement the Diagnosis service.
– The Multimedia NS introduced in Section 4.2, through the deployment of four VMs: two with the Hospital and Home NFs and the other two to implement the 4KHDStreaming and Telecare services. These four VMs can be seen detailed in the sequence diagram of Fig. 7.

Each of the previous VMs are composed of a single qcow2 image, which occupy about 2.5 GB each. Therefore, the RemoteCare NS is defined by three VMs and the Multimedia

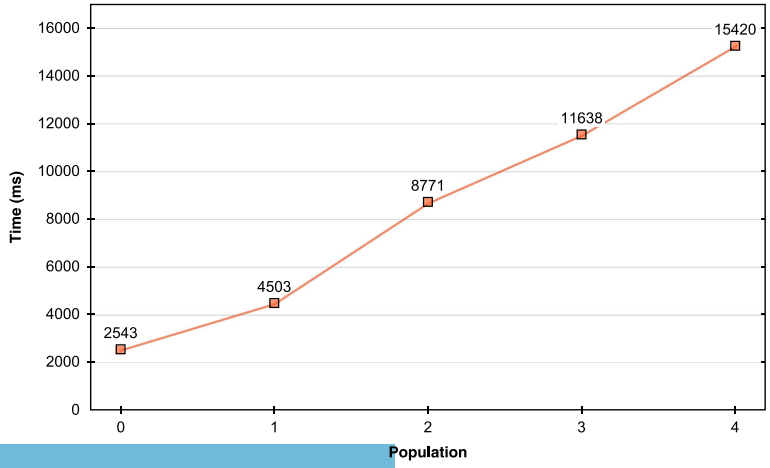

Fig. 8 Consistency checking time needed by the semantic reasoner

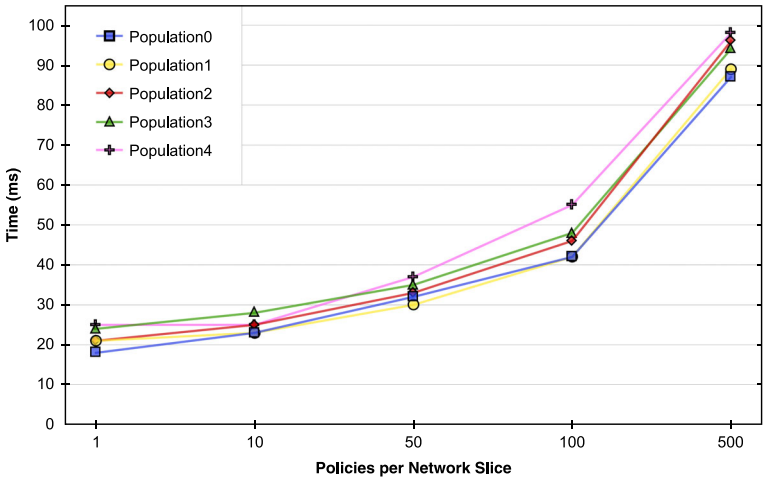

Fig. 9 Reasoning time for different populations and policies

NS by four, one qcow2 VM image per VxF. As experimental setting, the experiments presented below were conducted in a dedicated server with an Intel Xeon Processor E5-2697v4 at 3.6 GHz with 18 cores (36 threads) and 64 GB DDR4 of RAM, in which the different tools of Section 7.1 were integrated to deploy a fully virtualized network environment with SDN and NFV support: OpenStack as VIM, OpenDaylight as SDN Controller, and Open Baton as the Orchestrator. These tools were deployed in a single server on top of a virtualized platform in which each one of them will be running in detached processor cores, providing networking support separately as if it were deployed and configured in different physical nodes. It is worth highlighting that in this single server was implemented the fully virtualized network environment, in which the following experiments were conducted, which is based on the one designed and deployed as part of the 5G [12]. Further information can be found in its deliverable D2.4, publicly available at the [13], which deeply explains the configuration of this SDN/NFV environment. This deliverable also reports how the Wide Area Network (WAN) infrastructure, including the WIM for its management, is deployed and put into operation. The WIM proposed in SELFNET uses the SDN Controllers to provide dedicated and isolated virtual networks in the WAN, where functions can be applied by new SDN Apps in the framework by interfacing with the SDN Controller.

As a first experiment, Table 4 shows the performance times for the deployment of a single VxF with just one VM. The same configuration presented above was used, but without

Table 4 Performance times deploying a VxF with a single VM (in secs)

Flavor	Reserve resources	Configuration	Total time
m1.tiny	38.2937	94.4386	185.5987
m1.small	39.1944	97.4845	189.4063
m1.small	45.7710	109.9283	214.2710
m1.large	50.7711	124.6944	234.5060
m1.xlarge	62.0424	136.8837	258.7829

considering the needs of the two RemoteCare and Multimedia scenarios. This first test was performed to know how our infrastructure behaves when deploying a single VM (i.e. from Open Baton to the final deployment of the VxF), since the processes deploying virtualized elements will be different depending on the physical resources available. In this sense, the times shown in the table can be considered as base values, and can be used in next tests to see how they evolve as more elements are deployed.

Regarding Table 4, these tests were conducted by varying the flavors (compute, memory, and storage capacity) configured in OpenStack, which can be seen in the legend part of Fig. 10. The *Reserve resources* column refers to the time needed for reserving and allocating virtual resources in the NFVI through OpenStack, while *Configuration* denotes the precise time when the services in the VxF, which is running on a single VM, are configured for functioning through the associated VNFM. Finally, the last column of Table 4 shows the complete time required to deploy and configure the desired VxF.

As a result, the total time for the deployment of the same VM increases linearly as the flavor changes with better capacities, which corresponds mainly to the reserve of disk space for the deployment procedure; the increase in vCPU and RAM does not have a direct implication in those times. It is worth mentioning that this particular test can represent the execution of the intra-slice policy presented in Section 5.1, where the corrective measure is to add more computational resources into the RemoteCare NS: increasing the flavor and cloning the Diagnosis service to get more resources.

As a parallel test, the time results of the two scenarios discussed above (RemoteCare and Multimedia) are shown in Fig. 10 with respect to the deployment and instantiation of new VMs. The box plot outlines the times obtained after deploying from Open Baton all the VMs in 25 different tests separately for each scenario. It can be observed that there is a linear increase in time as more capacities of the resources (flavor) are being used. The provisioning time for the Network Slice commissioning, including the Diagnosis service of the RemoteCare scenario (3 VMs) or the 4KHDStreaming and Telecare services of the Multimedia one (4 VMs), takes on average 194.68 secs with a *m1.tiny* flavor and 288.91

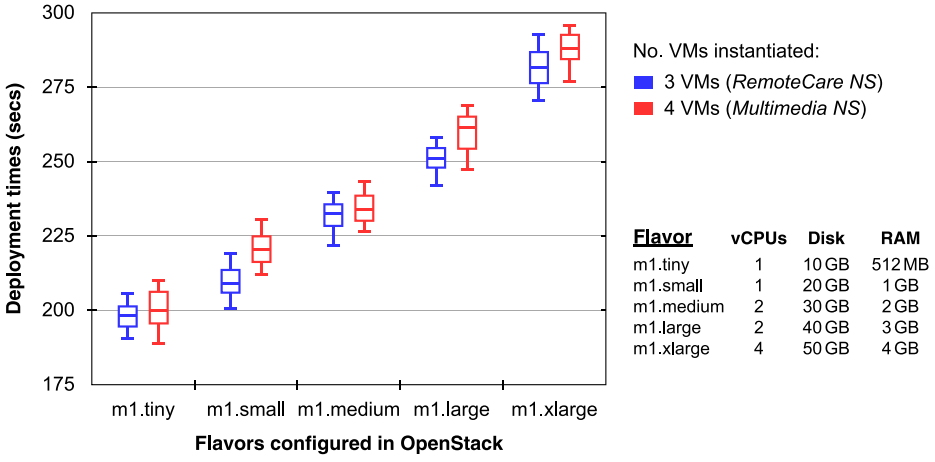

Fig. 10 Deployment times of three or four VMs to enable a given Network Slice, by varying the flavor capacities in OpenStack

secs with a *m1.xlarge* greater flavor, regardless of whether 3 or 4 VMs are being deployed. This fact indicates that deploying 3 VMs (RemoteCare NS) is similar in times than the ones required for 4 VMs (Multimedia NS). In spite of it, we can conclude that these performance times are fully acceptable to enable scenarios on remote healthcare supporting multimedia services, thus demonstrating the proper feasibility of our proposal.

After enabling a given NS for operation, as illustrated in the sequence diagram of Fig. 7, the last step is to activate the NS communications between the NFs, VMs, and services belonging to the NS (Step 20 in Fig. 7). To this end, the Orchestrator needs to install the corresponding OpenFlow rules in the virtual switches through the SDN Controller; OpenDaylight in our virtualized platform. In our environment virtual switches are integrated with the machines where the compute and networking nodes of OpenStack are running. Specifically, we have one switch per compute and another in the networking node. It is important to note that networks created in OpenStack make use of several ports of the previous switches. Regarding the generation of rules, the Automation Policy Engine, belonging to the Network Slice Administration layer, generates the OpenFlow rules. Once these rules are generated, the Orchestrator interacts with OpenDaylight through its northbound interface doing REST calls. Furthermore, it should also be emphasized that our environment can also offer support to the WIM, using the SDN Controllers to apply the OpenFlow rules to the distributed virtual switches and thus provide dedicated and isolated virtual networks in the WAN.

In order to measure the time required by this rule installation process, we conducted another test not only to extract and analyze times, but also to check scalability issues according to two network topology configurations: a *single network topology* with the four VMs of the Multimedia NS connected to the virtual switches, previously deployed by the previous test; and a more *complex topology* adding other 100 VMs implementing a lightweight qcow2 image of 1GB each for testing purposes. The average time after performing 100 tests is depicted in Fig. 11, where in addition to displaying the installation average times are also

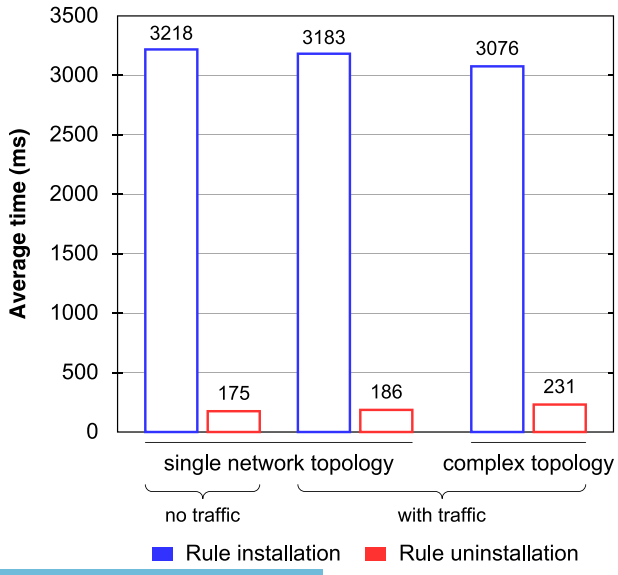

Fig. 11 Average time in the installation and uninstallation of OpenFlow rules in the virtual switches flow tables from the SDN Controller

shown the uninstallation time of rules. We also conducted these tests both with and without traffic, using for the former the Ostinato tool as network traffic generator implemented by [45]. It is also worth pointing out that the installation times also include checking that rules was added to the virtual switches flow tables successfully.

As shown in Fig. 11, the average time in installing/uninstalling rules in the virtual switches flow tables can be considered almost constant, regardless of the network topology used and the amount of network traffic injected. In general, the installation process takes just over 3 seconds, while the uninstall one about 200ms. Adding up this time in installing OpenFlow rules (about 3 seconds), the one related to the virtualized elements deployment of Fig. 10 (about 290 seconds on average in the worst case), and the one making decisions as detailed in Section 7.2 (about 15 seconds in the worst case), we can assume that they are quite reasonable times from the decision-making process to the final commissioning of a Network Slice: about 5 minutes in total. Yet, additional monitoring and predictive analytic procedures would be necessary to have a complete autonomic computing framework.

The performance time results obtained and discussed in previous sections can be considered quite satisfactory outcomes. They perfectly fit into the Key Performance Indicators (KPIs) established by the [16], which have been redefined and trimmed by certain projects funded by the European Commission's H2020 program. For example, the [14] consortium defined in its deliverable D3.1 that the deployment times of NFV resources must be below 25 minutes, defined as KPI 8 in that deliverable, reducing very considerably the initial expected creation time "from 90 hours to 90 minutes" as established by European 5G initiatives such as the [11].

Finally, it is worth pointing out that other administrative tasks can be required by network operators and service providers before enforcing the operations detailed in this article, such as *authorization* to launch new resources and services, *access control* to the different tools that allow such provision, *service level agreement* establishing customer-provider relationships as defined by the OSS/BSS layer (see Fig. 5), just to name a few.

8 Conclusion and future work

This article has presented a solution capable of managing the complete life cycle of Network Slices through the combination of the SDN/NFV techniques for dynamic service provision, taking into consideration the specific requirements in an eHealth scenario with multimedia services. The proposed architecture can determine when, what, and how to dynamically orchestrate the resources and the services to meet the requirements established by given scenarios. To this end, we have proposed a formal Network Slicing information model based on ontologies in which all elements interacting with the Network Slices are formally represented. These ontologies are managed by a novel architecture that combines the SDN/NFV techniques, and a policy-based system that defines such policies to control the Network Slices as well as their resources dynamically. Two types of policies have been defined to manage the elements making up the current Network Slice (intra-slice policies) and change that current Network Slice for a new one that meets the established requirements (inter-slice policies). Conducted experiments have demonstrated the feasibility of the proposed architecture, which were performed in two scenarios oriented to future remote healthcare with multimedia services.

As future work, we plan to extend the current proposal with monitoring and predictive analytic procedures to collect and analyze the diverse metrics. This work will allow

us to close the current proposal with an autonomic computing framework to provide a complete functionality from monitoring and sensing to the actuation layers. In addition, the authors' intention is continue working in patients monitoring to collect more valuable information from heart monitors, infusion pumps, and wearables, among others, depending on the diseases to be analyzed, and thus provide better treatment to such patients in remote.

Acknowledgments This work has been supported by a Séneca Foundation grant within the Human Resources Researching Postdoctoral Program 2018; by the Irish Research Council, under the government of Ireland post-doc fellowship (grant GOIPD/2018/466); and by a post-doctoral INCIBE grants within the "Ayudas para la Excelencia de los Equipos de Investigación Avanzada en Ciberseguridad" Program, with code INCIBEI-2015-27352.

References

1. Afolabi I, Taleb T, Samdanis K, Ksentini A, Flinck H (2018) Network slicing and softwarization: A survey on principles, enabling technologies, and solutions. IEEE Commun Surv Tutorials 20(3):2429–2453
2. Asorey Cacheda R, Castro García D, Cuevas A, González Castaño FJ, Herrero Sánchez J, Koltsidas G, Mancuso V, Moreno Novella JI, Oh S, Pantò A (2007) QoS requirements for multimedia services. In: Resource Management in Satellite Networks, Springer, pp 67–94
3. Da Silva I, Mildh G, Kaloxylos A, Spapis P, Buracchini E, Trogolo A, Zimmermann G, Bayer N (2016) Impact of network slicing on 5G radio access networks. In: 2016 European Conference on Networks and Communications, pp 153–157
4. De Turck F, Boutaba R, Chemouil P, Bi J, Westphal C (2015) Guest editors' introduction: Special Issue on efficient management of SDN/NFV-based systems. IEEE Trans Netw Serv Manag 12(2):114–116
5. Distributed Management Task Force Inc (2018) The CIM standard: Common Information Model. [retrieved: Jan. 2019]. https://www.dmtf.org/standards/cim/
6. ETSI (2015) Network Functions Virtualisation (NFV); Infrastructure Overview. ETSI GS NFV-INF 001 V1. 1:1
7. Ferrus R, Sallent O, Perez-Romero J, Agusti R (2018) On 5G radio access network slicing: Radio interface protocol features and configuration. IEEE Commun Mag 56(5):184–192
8. Foukas X, Patounas G, Elmokashfi A, Marina MK (2017) Network slicing in 5G: Survey and challenges. IEEE Commun Mag 55(5):94–100
9. Gavras A, Denazis S, Tranoris C, Hrasnica H, Barros Weiss M (2017) Requirements and design of 5G experimental environments for vertical industry innovations. In: 2017 Global Wireless Summit, pp 165–169
10. Gutz S, Story A, Schlesinger C, Foster N (2012) Splendid isolation: A slice abstraction for software-defined networks. In: 1st Workshop on Hot Topics in Software Defined Networks, pp 79–84
11. H2020 Euro-5G Project (2017) Deliverable 2.6: Final report on programme progress and KPIs. [retrieved: Jan. 2019]. https://5g-ppp.eu/wp-content/uploads/2017/10/Euro-5G-D2.6_Final-report-on-programme-progress-and-KPIs.pdf
12. H2020 SELFNET Project (Unknown Month 2015) SELFNET - Framework for Self-Organized Network Management in Virtualized and Software Defined Networks. [retrieved: Jan. 2019]. https://selfnet-5g.eu/
13. H2020 SELFNET Project (2016) Deliverable 2.4: Portable testbed to execute virtualized NFV-based and SDN-based scenarios. [retrieved: Jan. 2019]. https://doi.org/10.18153/SLF-671672-D2_4

14. H2020 SoftFIRE Project (2017) Deliverable 3.1: KPIs for evaluating and assessing the features of the testbed. [retrieved: Jan. 2019]. https://www.softfire.eu/wp-content/uploads/SoftFIRE-D3.1-KPIs-Testbed-evaluation-v3.1.pdf
15. Horrocks I, Patel-Schneider PF, Boley H, Tabet S, Grosof B, Dean M (2004) SWRL: A semantic web rule language combining OWL and RuleML, W3C Submission. [retrieved: Jan. 2019]. https://www.w3.org/Submission/SWRL/
16. (https://5g-ppp.eu/kpis/) Key Performance Indicators (KPIs). [retrieved: Jan. 2019]
17. Huertas Celdrán A, Gil Pérez M, García Clemente FJ, Martínez Pérez G (2017) Preserving patients' privacy in health scenarios through a multicontext-aware system. Ann Telecommun 72(9-10):577–587
18. Huertas Celdrán A, Gil Pérez M, García Clemente FJ, Ippoliti F, Martínez Pérez G (2018) Policy-based network slicing management for future mobile communications. In: 5th IEEE International Conference on Software Defined Systems, pp 153–159
19. Huertas Celdrán A, Gil Pérez M, García Clemente FJ, Martínez Pérez G (2018) Policy-based management for green mobile networks through software-defined networking. Mobile Networks and Applications, In Press. https://doi.org/10.1007/s11036-016-0783-8
20. Jiang M, Condoluci M, Mahmoodi T (2016) Network slicing management & prioritization in 5G mobile systems. In: 22th European wireless conference on european wireless, vol 2016, pp 1-6
21. Li Z, O'Brien L, Zhang H, Cai R (2012) On a catalogue of metrics for evaluating commercial cloud services. In: ACM/IEEE 13th International Conference on Grid Computing, pp 164–173
22. Li X, Samaka M, Chan HA, Bhamare D, Gupta L, Guo C, Jain R (2017) Network slicing for 5G: Challenges and opportunities. IEEE Internet Comput 21(5):20–27
23. Makhijani K, Qin J, Ravindran R, Geng L, Qiang L, Peng S, de Foy X, Rahman A, Galis A (2017) Network slicing use cases: Network customization and differentiated services, IETF Internet-Draft draft-netslices-usecases-02
24. Michalopoulos DS, Doll M, Sciancalepore V, Bega D, Schneider P, Rost P (2017) Network slicing via function decomposition and flexible network design. In: 2017 IEEE 28th annual international symposium on personal, indoor, and mobile radio communications, pp 1–6
25. Modi KJ, Kapadia N (2019) Securing healthcare information over cloud using hybrid approach. In: Progress in advanced computing and intelligent engineering, pp 63–74
26. Motik B, Patel-Schneider PF, Parsia B (eds.) (2012) OWL 2 web ontology language: Structural specification and functional-style syntax, 2nd edn. https://www.w3.org/TR/owl2-syntax/
27. NGMN Alliance (2016) Description of network slicing concept, NGMN 5G P1 Deliverable. [retrieved: Jan. 2019]. https://www.ngmn.org/fileadmin/user_upload/161010_NGMN_Network_Slicing_framework_v1.0.8.pdf
28. OASIS (2017) Topology and orchestration specification for cloud applications (TOSCA). [retrieved: Jan. 2019]. https://www.oasis-open.org/committees/tosca/
29. Open Baton Project (2017) A ETSI NFV compliant MANO framework. [retrieved: Jan. 2019]. https://openbaton.github.io/
30. Open Networking Foundation (2016) TR-526 applying SDN architecture to 5G slicing. [retrieved: Jan. 2019]. https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/Applying_SDN_Architecture_to_5G_Slicing_TR-526.pdf
31. OpenDaylight Project (2016) An open source SDN platform. [retrieved: Jan. 2019]. https://www.opendaylight.org/
32. OpenStack Project (2016) Open source software for creating private and public clouds. [retrieved: Jan. 2019] https://www.openstack.org/
33. Ordonez-Lucena J, Ameigeiras P, Lopez D, Ramos-Muñoz JJ, Lorca J, Folgueira J (2017) Network slicing for 5G with SDN/NFV: Concepts, architectures and challenges. IEEE Commun Mag 55(5):80–87
34. Qiang L, Martinez-Julia P, Geng L, Dong J, Makhijan K, Galis A, Hares S, Kuklinski S (2017) Gap analysis for transport network slicing, IETF Internet-Draft draft-qiang-netslices-gap-analysis-01
35. Ravindran R, Chakraborti A, Amin SO, Azgin A, Wang G (2017) 5G-ICN: Delivering ICN services over 5G using network slicing. IEEE Commun Mag 55(5):101–107
36. Raza MT, Lu S (2017) Enabling low latency and high reliability for IMS-NFV. In: 13th international conference on network and service management, pp 1–9
37. Richart M, Baliosian J, Serrati J, Gorricho JL, Agüero R, Agoulmine N (2017) Resource allocation for network slicing in WiFi access points. In: 2017 13th international conference on network and service management, pp 1–4

38. Riegel M (2017) Key concepts of network instantiation, IEEE 802.1 OmniRAN Task Group. [retrieved: Jan. 2019]. https://mentor.ieee.org/omniran/dcn/16/omniran-16-0071-01-CF00-key-concepts-of-an-instantiation.pptx
39. Rost P, Mannweiler C, Michalopoulos DS, Sartori C, Sciancalepore V, Sastry N, Holland O, Tayade S, Han B, Bega D, Aziz D, Bakker H (2017) Network slicing to enable scalability and flexibility in 5G mobile networks. IEEE Commun Mag 55(5):72–79
40. Schneider P, Mannweiler C, Kerboeuf S (2018) Providing strong 5G mobile network slice isolation for highly sensitive third-party services. In: IEEE wireless communications and networking conference, pp 1–6
41. Sherwood R, Chan M, Covington A, Gibb G, Flajslik M, Handigol N, Huang T, Kazemian P, Kobayashi M, Naous J, Seetharaman S, Underhill D, Yabe T, Yap K, Yiakoumis Y, Zeng H, Appenzeller G, Johari R, McKeown N, Parulkar G (2010) Carving research slices out of your production networks with OpenFlow. SIGCOMM Comput Commun Rev 40(1):129–130
42. Sirin E, Parsia B, Cuenca Grau B, Kalyanpur A, Katz Y (2007) Pellet: A practical OWL-DL reasoner. Web Semant Sci Serv Agents World Wide Web 5(2):51–53
43. Solozabal R, Sanchoyerto A, Cava M, Blanco B, Khalife H, Bouet M, Lavaux D, Kafetzakis E (2018) Providing mission-critical services over 5G Radio Access Network. In: 14th IFIP WG 12.5 international conference on artificial intelligence applications and innovations, pp 520–530
44. Soon-Shiong P, Kupwade-Patil H, Seshadri R, Witchey NJ (2018) Homomorphic encryption in a healthcare network environment, system and methods. US Patent App 15(/727):494
45. Srivats P (2018) Ostinato packet generator. [retrieved: Jan. 2019]. https://ostinato.org/
46. (2016) Protégé: A free, open source ontology editor and knowledge-base framework. [retrieved: Jan. 2019]. https://protege.stanford.edu/
47. Taleb T, Mada B, Corici MI, Nakao A, Flinck H (2017) PERMIT: Network slicing for personalized 5G mobile telecommunications. IEEE Commun Mag 55(5):88–93
48. University of Murcia (2018) Complete definition of the Network Slicing ontologies. [retrieved: Jan. 2019]. http://selfnet.inf.um.es/nsontology/
49. W3C Recommendation (2013) SPARQL 1.1 query language for RDF. [retrieved: Jan. 2019]. https://www.w3.org/TR/rdf-sparql-query/
50. Zhang H, Liu N, Chu X, Long K, Aghvami AH, Leung VCM (2017) Network slicing based 5G and future mobile networks: Mobility, resource management, and challenges. IEEE Commun Mag 55(8):138–145
51. Zhou X, Li R, Chen T, Zhang H (2016) Network slicing as a service: Enabling enterprises' own software-defined cellular networks. IEEE Commun Mag 54(7):146–153

Alberto Huertas Celdrán is Researcher Associate in the Telecommunications Software & Systems Group, Waterford Institute of Technology, Ireland. His scientific interests include security of interoperable clinical environments and medical devices, Software-Defined Networking (SDN), semantic technology, and policy-based context-aware systems. Huertas Celdrán received M.Sc. and Ph.D. degrees in Computer Science from the University of Murcia, Spain. He has published more than 30 papers in national and international conference proceedings, magazines and journals.

Manuel Gil Pérez is Research Associate in the Department of Information and Communication Engineering of the University of Murcia, Spain. His research primarily focuses on cybersecurity, trust management, and security operations in highly dynamic scenarios. Gil Pérez received an M.Sc. and Ph.D. (Hons) degrees in Computer Science from the University of Murcia. He is coauthor of 60+ scientific publications in journals, magazines and conference papers.

Félix J. García Clemente is Associate Professor of Computer Networks in the Department of Computer Engineering of the University of Murcia, Spain. His research interests include security and management of distributed communication networks. García Clemente received M.Sc. and Ph.D. degrees in Computer Science from the University of Murcia. He has published more than 100 papers in national and international conference proceedings, magazines and journals.

Fabrizio Ippoliti received M.Sc. and Ph.D. degrees in Computer Science from the University of Camerino, Italy. His main scientific activity is devoted to security infrastructures, Software-Defined Networking (SDN) and Virtual Network Function (VNF) techniques, as well as trust modeling and management for cloud computing.

Gregorio Martínez Pérez is Full Professor in the Department of Information and Communication Engineering of the University of Murcia, Spain. His scientific activity is mainly devoted to cybersecurity, privacy and networking. Martínez Pérez received M.Sc. and Ph.D. degrees in Computer Science from the University of Murcia. He has published more than 160 papers in national and international conference proceedings, magazines and journals. He has already supervised 10 PhD students, several of them recognized with honours.

Affiliations

Alberto Huertas Celdrán[1] · Manuel Gil Pérez[2] · Félix J. García Clemente[3] · Fabrizio Ippoliti[4] · Gregorio Martínez Pérez[2]

Manuel Gil Pérez
mgilperez@um.es

Félix J. García Clemente
fgarcia@um.es

Fabrizio Ippoliti
fabrizio.ippoliti@unicam.it

Gregorio Martínez Pérez
gregorio@um.es

1 Telecommunications Software, Systems Group, Waterford Institute of Technology, X91 K0EK Waterford, Ireland

2 Departamento de Ingeniería de la Información y las Comunicaciones, University of Murcia, 30071 Murcia Spain

3 Departamento de Ingeniería y Tecnología de Computadores, University of Murcia, 30071 Murcia, Spain

4 Computer Science Division, School of Science and Technology, University of Camerino, 62032 Camerino, Italy