Computer Standards & Interfaces 54 (2017) 203–215

Contents lists available at ScienceDirect

Computer Standards & Interfaces

journal homepage: www.elsevier.com/locate/csi

Network service orchestration standardization: A technology survey

Charalampos Rotsos[a,*], Daniel King[a], Arsham Farshad[a], Jamie Bird[a], Lyndon Fawcett[a], Nektarios Georgalas[b], Matthias Gunkel[c], Kohei Shiomoto[d], Aijun Wang[e], Andreas Mauthe[a], Nicholas Race[a], David Hutchison[a]

[a] Infolab21, School of Computing and Communications, Lancaster University, United Kingdom
[b] British Telecom, United Kingdom
[c] Deutsche Telekom, Germany
[d] NTT, Japan
[e] China Telecom, China

ARTICLE INFO

Keywords:
Network Service
Orchestration
SDN
NFV
Standardization

ABSTRACT

Network services underpin operator revenues, and value-added services provide income beyond core (voice and data) infrastructure capability. Today, operators face multiple challenges: a need to innovate and offer a wider choice of value-added services, whilst increasing network scale, bandwidth and flexibility. They must also reduce operational costs, and deploy services far faster - in minutes rather than days or weeks.

In the recent years, the network community, motivated by the aforementioned challenges, has developed production network architectures and seeded technologies, like Software Defined Networking, Application-based Network Operations and Network Function Virtualization. These technologies enhance the highly desired properties for elasticity, agility and cost-effectiveness in the operator environment. A key requirement to fully exploit the benefits of these new architectures and technologies is a fundamental shift in management and control of resources, and the ability to orchestrate the network infrastructure: coordinate the instantiation of high-level network services across different technological domains and automate service deployment and re-optimization.

This paper surveys existing standardization efforts for the orchestration - automation, coordination, and management - of complex set of network and function resources (both physical and virtual), and highlights the various enabling technologies, strengths and weaknesses, adoption challenges for operators, and areas where further research is required.

1. Introduction

> Flexibility, agility and automation and a much faster time-to-market cycle, where the latter is something that we, as operators, lack today
> (Christos Kolias, Network Architect, Orange [1]).

Network services are the primary value-added products for Network Operators (operators), enabling them to monetize their infrastructure investments. Operator service portfolios cover a wide range of functionalities, spanning from basic Internet connectivity services, such as IPTV delivery, to highly-available and secure connectivity between business sites. This operator business model has been highly successful, their user base continuously expands [2], while new services are adopted by end-users.

As a direct consequence, network infrastructures have grown significantly in the recent years and operators face significant challenges maintaining high revenues, while supporting innovative new network services. On the one hand, traffic volumes increase exponentially [3] and forces operators to upgrade infrastructures frequently. Additionally, the established service management model relies extensively on manual device reconfiguration by the network engineers, coordinated through Operational Support Systems (OSS), while link over-provision is used to enforce SLAs. Effectively, the predominant service management model incurs significant capital (CAPEX) and operational expenditures (OPEX) for the operator [4]. On the other hand, network infrastructures employ a widening range of heterogeneous technologies to support the diverse characteristics and dynamic demands of residential and enterprise network services. Unfortunately,

* Corresponding author.
E-mail addresses: h.rotsos@gmail.com, cr409@cl.cam.ac.uk (C. Rotsos).

http://dx.doi.org/10.1016/j.csi.2016.12.006
Received 15 April 2016; Received in revised form 4 November 2016; Accepted 31 December 2016
Available online 07 February 2017

the control and management interfaces of the relevant technologies do not keep abreast with the requirements of network applications for fluid and dynamic control. The different technological domains and layers exhibit significant interface proliferation, while vertical control integration in network devices impairs management flexibility and responsiveness. As a result, the futuristic vision of network operators to provide service-oriented control interfaces to end-user applications, still remains unfulfilled.

These limitations have motivated the network and systems community to develop new paradigms and architectures which improve network infrastructure flexibility, agility, programmability and elasticity and ensure low OPEX. Recent network paradigms, like Software Defined Networking (SDN) and Application-based Network Operations (ABNO), promote control convergence across network layers and logical centralization of network infrastructure management through the specification of common device control interfaces. In parallel, the Network Function Virtualization (NFV) paradigm promotes the ”*softwarization*” and virtualization of network functions, in order to enable data plane processing with similar elasticity, scalability and resilience available in cloud environments. Furthermore, new network architectures including Service Functions Chaining (SFC) and Segment Routing (SR), simplify service deployment and allow seamless integration of traffic-engineered (deterministic) network services and network policy.

To capitalize on the fluidity of these novel networking paradigms and architectures, operators a require new control and management system, capable to *orchestrate* the different technologies and resource types available in modern network infrastructures. These systems are responsible to converge control and management heterogeneity between technologies, in an effort to synthesize innovative service-oriented interfaces, and enable autonomous and automated service deployment and adaptation. The development of service orchestration architectures and interfaces has been accelerating, but since each vendor typically develops its own protocols and mechanisms, integration remains a challenge. Towards the goal for automated, flexible and cost-effective service orchestration, interoperability and standardization play a crucial role for its success.

This paper surveys standardization efforts towards enabling network service orchestration from an operator perspective. To elaborate on available interfaces, standards and recommendations we follow a top-down approach. We begin with a definition of the document terminology, and we elaborate on the network service orchestrator requirements and objectives from the perspective of four of the world's largest and complex network operators —British Telecom, Deutsche Telekom, NTT and China Telecom — (Section 2). Furthermore, to motivate our discussion on network services, we present the design and requirements of three popular network service use cases, namely Radio Access Network and Mobile Evolved Packet Core connectivity services and end-to-end content distribution service (Section 3). We then elaborate on the capabilities and interfaces of the predominant network (Section 4) and function (Section 5) management and control architectures. Finally, we discuss the future directions for network orchestration standardization efforts (Section 6) and conclude this paper (Section 7).

2. What is network service orchestration?

2.1. Terminology

A *network service* is a high-level network functionality that generates business value for customers and/or the operator. Network services are typically represented as directed graphs, where the nodes of the graph represent low-level network functions and the directed edges describe ordering and connectivity.

A *network or service function (NF)* is a specialized network element, designed to efficiently perform a restricted set of low-level operations on traffic. An NF can manipulate traffic at multiple layers of the protocol stack and it is common to manipulate packets traversing the network, as well as terminate network flows. Virtual software instances, such as a Broadband Network Gateway (vBNG) or IP Multimedia Subsystem (vIMS) running on a virtual machine, or specialized physical hosts, such as hardware load-balancers, are both common approaches to realize NFs. Furthermore, virtualization allows instantiation of multiple NFs on a single physical node, while a single physical node can potentially support the instantiation of multiple different NF types. Finally, NFs predominantly are designed to modify network traffic, but passive monitoring NFs are equally popular, such as intrusion detection systems.

A *Service Orchestrator* is a control system for the provision, management and re-optimization of network services. Effectively, a service orchestrator receives network service requests from individual applications, service consumers and the operator. Based on the received service requests, the available infrastructure resources and the topological properties of the underlying network, the orchestrator is responsible to define and execute a deployment plan that fulfills the NF and connectivity requirements of each service. In parallel, the service orchestrator monitors the performance of all services and dynamically adjusts the infrastructure configuration to continuously ensure the performance guarantees and cost goals.

Service Orchestration aims to support a wide range of infrastructure technologies and resource types and depends on technical standards to broaden its applicability. A technical standard reflects an established set of requirements or norms to precise technical systems. They are typically formal documents that establish uniform engineering or technical criteria, procedures, protocols and practices. This survey paper investigates the myriad of SDN and NFV standards (both formal and de-facto) across a range of Standards Development Organizations (SDO), and rapidly expanding environment of Open Source software projects. Typically, the impedance mismatch between SDOs and Open Source is at least 2:1 (two years to a paper standard versus one year to a product that creates a de-facto standard) [5].

2.2. Requirements

A Service orchestration is a complex high-level control system and relevant research efforts have proposed a wide range of goals for a service orchestrator. We identify the following functional properties:

Coordination: Operator infrastructures comprise of a wide range of network and computation systems providing a diverse set of resources, including network bandwidth, CPU and storage. Effective deployment of a network service depends on their coordinated configuration. The network manager must provision network resources and modify the forwarding policy of the network, to ensure ordering and connectivity between the service NFs. This process becomes complex when considering the different control capabilities and interfaces across network technologies found in the metropolitan, access and wide area layers of the operator network. Furthermore, the network manager must configure the devices that will host the service NFs, either in software or hardware. The service orchestrator is responsible for abstracting the management and configuration heterogeneity of the different technologies and administrative domains [6,7].

Automation: Existing infrastructures incur significant operational workload for the configuration, troubleshooting and management of network services. Network technologies typically provide different configuration interfaces in each network layer and require manual and repetitive configuration by network managers to deploy a network service [8]. In addition, vertical integration of network devices requires extensive human intervention to deploy and manage a network service in a multi-vendor and multi-technology environment. A key goal for service orchestration is to minimize human intervention during the deployment and management of network services. Efforts in programmable network and NFV control, like SDN, ABNO and ETSI NFV

MANO, provide low-level automation capabilities, which can be exploited by the service orchestrator to synthesize high-level automation service deployment and management mechanisms [9].

Resource provision and monitor: The specification of network services contain complex SLA guarantees, which perplex network management. For example, allocating resources, which meet service delivery guarantees, is an NP-hard problem from the perspective of the operator and the re-optimization of a large network can take days. In parallel, existing service deployment approaches rely on static resource allocations and require resource provision for the worst-case service load scenarios. A key goal for service orchestration is to enable dynamic and flexible resource control and monitoring mechanisms, which converge resource control across the underlying technologies and abstract their heterogeneity [10,11].

Efforts towards service orchestration are still limited. Relevant architecture and interface specifications define mechanisms for effective automation and programmability of individual resource types, like the SDN and ABNO paradigms for network resources and the NFV MANO for compute and storage resources. Nonetheless, these architectures remain low-level and provide partial control over the infrastructure towards service orchestration. Service orchestration initiatives from network operators and vendors [12,13] propose the development of a new orchestration layer above and beyond the existing individual control mechanisms which will capitalize on their low-level automation and flexibility capabilities to support a service-oriented control abstraction exposed to the OSS/BSS, as depicted in Fig. 1. In terms of network control, the service orchestrator can access low-level forwarding interfaces, as well as high high-level control interfaces implementing standardized forwarding control mechanisms, like Segment Routing and Service Function Chain, through the network controller. In parallel, NF management across the operator datacenters can be achieved through a dual-layer control and management stack, as suggested by relevant NF management architectures. The lower layer contains the Virtual Infrastructure Manager (VIM), which manages and configures the virtualization policy of compute and storage resources. The top layer contains the VNF Manager (VNFM) responsible for the configuration, control and monitor of individual NFs. The service orchestrator will operate on top of these two management services (network and IT, see Fig. 1) and will be responsible for exploiting their functionality to provide network service delivery, given the policy of the operator, channeled through the OSS. The effectiveness of the service orchestrator highly depends on the granularity and flexibility of the underlying control interfaces. This paper surveys standardization efforts for infrastructure control in an effort to discuss

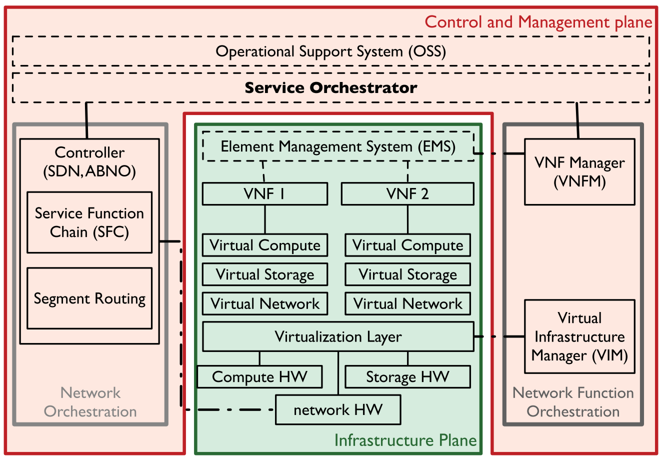

Fig. 1. An architectural model for service orchestration in operator infrastructure. The orchestrator uses the interfaces exported by the network controller and the VNF Manager to control the deployment, management, configuration and troubleshooting of network services.

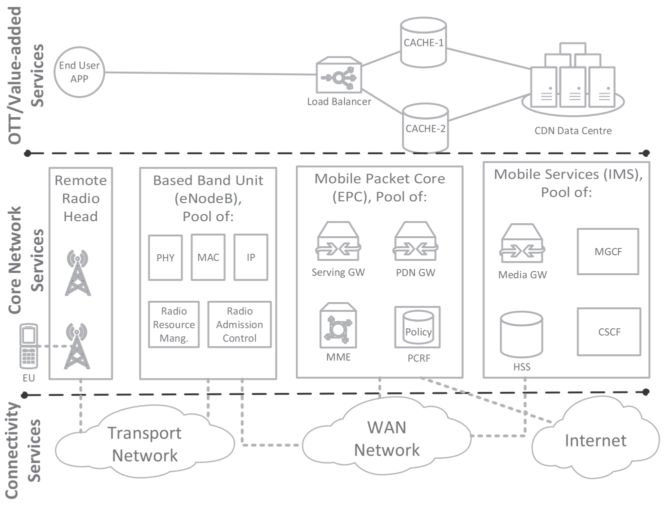

Fig. 2. An aggregate view of the functional blocks which deliver CDN and other value-added services to a mobile network.

the existing opportunities and challenges towards service orchestration.

3. Network services

Network services enable a wide range of value-added functionalities for operators and users across all layers of the infrastructure. This section presents three popular network services to identify control requirements for a service orchestrator. Specifically, we elaborate on the architecture of mobile radio access and core networks, followed by a discussion on CDN services as an example of a value-added service.

Fig. 2 depicts the abstract view of the service chain of the discussed services, along with their functional block. The figure illustrates three layers of network services: connectivity services provided by the network infrastructure; core network services that provide communication and value-added services to end-users of the network; and a top application layer, which delivers an application service to the end-user.

3.1. Radio access network (RAN)

The 3G standards split the mobile RAN in two functional blocks: the *Remote Radio Head (RRH)*, which receives and transmit the wireless signal and applies the appropriate signal transformations and amplification, and the *Base Band Unit (BBU)*, which runs the MAC protocol and coordinates neighboring cells. The channel between these two entities has high bandwidth and ultra-low latency requirements and the two systems are typically co-located in production deployments. Nonetheless, this design choice increases the operator cost to deploy and operate its RAN. BBUs are expensive components which increase the overall acquisition cost of a base station, while the BBU cooling requirements makes the RAN a significant contributor to the aggregate power consumption of the operator [14].

Recent trends in RAN design separate the two components, by moving the BBU to the central office of the operator; an architectural paradigm commonly termed Cloud-RAN (C-RAN). C-RAN significantly reduces deployment and operational costs and improves elasticity and resilience of the RAN. In parallel, the centralization of multiple RRHs under the control of a single BBU improves resource utilization and cell handovers, and minimizes cell-interference. Currently multiple interfaces, architectures and testbeds provide the technological capabilities to run and test C-RAN systems [15,16], while vendors currently provide production-ready virtualized BBU appliances [17]. In addition, novel control abstractions can converge RAN control with underlying transport technologies and enable flexible deployment strategies [18].

A challenge for C-RAN architectures is the high multi-Gb bandwidth requirements and strict sub-milliseconds latency and jitter demands for the links between the RRH and the datacenter [19]. These connectivity guarantees exhibit significant variability (from a few Mb to 30 Gb) within the course of a day, reflecting the varying loads of mobile cell, as well as the signal modulation and channel configuration. To provide flexible and on-demand front-haul connectivity with strong latency guarantees, operators require novel orchestration mechanisms supporting dynamic and multi-technology resource management. In addition, effective RAN virtualization requires a framework for the management and monitoring of BBU instances to provide service resiliency. The service orchestrator can monitor the performance of the BBU VNF instances and adjust the compute resource allocation, the VNF replication degree and the load distribution policy. In parallel, the orchestrator can improve front-haul efficiency by mapping the connectivity requirements between the BBU and the RRH in network resource allocation policy.

The 3rd Generation Partnership Project (3GPP) is actively exploring the applicability of NFV technologies on a range of mobile network use-cases, like fault-management and performance monitoring, and has defined a set of management requirements in the RAN, the Mobile Core Network and the IP Multimedia Subsystem (IMS) [20]. In parallel, the 5G Public Private Partnership (5G PPP), within its effort to standardize the technologies and protocols for the next generation communication network defines end-to-end network service orchestration as a core design goal [21].

3.2. Evolved packet core (EPC)

Evolved Packet Core (EPC) is a network architecture for the core network of mobile operators, introduced in the 4G standards. It converges voice and data traffic in a single IP-based infrastructure. EPC comprises of different functional elements providing the core mobile network services. The EPCs main functional blocks are presented in Fig. 2. The Service Gateway (SGW) is the gateway terminating the interface toward the RAN. Packet Data Network Gateway (PGW) is the gateway to Packet Delivery Network (PDN) and enforces per-user packet filtering, policing/shaping rate and traffic accounting. The Mobility Management Entity (MME) and Policy and Charging Rules Functions (PCRF) are acting as controllers for mobility and billing functions. Furthermore, the IMS provides signaling for the establishment and termination of end-to-end packet-based multimedia services, like Voice over LTE (VoLTE). These functions are currently delivered using expensive integrated network devices, which provide limited modularity and interoperability between vendors. Thus, ensuring EPC service delivery guarantees during peak times, can be achieved only during the network planning phase through network and function over-provision. Furthermore, running multiple logical networks, each providing different performance guarantees and functionalities, over a single physical infrastructure, a key functionality for 5G technologies termed *network slicing*, will require extensive virtualization of the key EPC functions [22].

Multiple studies have argued for the softwarization of the key EPC functional blocks and the introduction of programmability in the EPC network control. SoftAir [23] is a software-defined architecture for next generation mobile networks using network and function virtualization paradigms for both the EPC and the RAN. Open5GCore [24] is another effort toward the cloudification of the EPC. Effectively, the framework provides an LTE protocol stack and supports uniform and distributed control plane. Furthermore, carrier-grade IMS VNF products are readily available from different vendors [25]. Finally, both IMS and EPC services are primary use cases for the European Telecommunications Standards Institute (ETSI) NFV Industrial Specification Group (ISG) [26].

3.3. Content delivery network (CDN)

CDN services provide efficient distribution of static content on behalf of third-party Internet applications [27]. They rely on a well-provisioned and highly-available network of cache servers and allow end-users to retrieve static content with low latency by automatically redirecting them to an appropriate cache server, based on the user location, the caching policy and cache load. CDN traffic currently constitutes a large portion of the operator traffic volumes and providers, like Akamai, serve 15–30% of the global Internet traffic [28].

The CDN service chain is simple and consists of a load-balancing function and a cache function, as depicted in Fig. 2. The greatest challenge in the deployment of such a service is the aggregate network data volumes of the service and the large number of network end-points. As a result, temporal variations in CDN traffic patterns can have a dramatic effect on the traffic matrix of the operator and affect Internet service delivery. In parallel, CDN-ISP integration lacks support for dynamical resource provision, in order to gracefully manage the dynamic traffic patterns. Connectivity relies on fixed-capacity peering relationships through popular IXPs or CDN-operated peering locations [29], which must be provisioned for the worst-case scenario.

The current design of CDN services introduces an interesting joint optimization problem between operators and CDN service providers. A CDN service bring content closer to the user and enable dynamic deployment of caching NFs in the central offices of the operator and enforce network resource guarantees. The service can provide sufficient elasticity for the CDN caching layer, while the ISP can reduce core network load. Similar approaches have been proposed in the context of mobile operators, mobile CDN emerged to faster access to mobile apps, facilitate mobile video streaming and supporting dynamic contents [30,31]. In parallel, new network control architectures based on SDN and NFV principles enable CDN services to localize users and offload the redirection task in the network forwarding policy [32,33]. These approaches provide an innovative environment to improve CDN functionality, but require a flexible control mechanism to integrate CDN services and infrastructures. A service orchestrator can autonomously adapt the CDN service deployment plan to the CDN load characteristics, using a policy specification from the CDN provider. In parallel, the orchestrator can monitor traffic volumes to infer content locality and hotspot development and deploy NF caches close to the end-user to improve latency and network efficiency.

4. Network orchestration standardization

Modern operator infrastructures contain a wide range of technologies across all network layers. Typically, the network of an operator is separated into multiple control domains (access, metropolitan and core), each using different network technologies, control interfaces and implementing forwarding policy with diverse goals [34]. Management, configuration and troubleshooting processes rely extensively on human intervention, to translate high-level connectivity goals into individual device configurations, while service deployment is designed in paper by network managers. As a result, service lead-times for new services can take up to a few months [35], with the majority of this time spent in the design and configuration of network infrastructures.

The inflexibility and limited automation in the network infrastructure has motivated the development of new control and management architectures and protocols. An important design goal for these new networking paradigms is standardization and openness of interfaces, in order to overcome the existing inter-operability limitations created by the vertical integration of network devices. In this section, we elaborate on two recent and highly successful control architectures; SDN (Section 4.1) and ABNO (Section 4.2). Such paradigms provide the required low-level control interfaces to effectively deploy services across an operator network and to control network resources. Our presentation focuses on the architecture of the respective paradigms and elaborates on the standardization efforts for the interfaces exposed to the service orchestrator.

4.1. Software defined networking (SDN)

SDN [36] is a recent network paradigm aiming for automated, flexible and user-controlled network forwarding and management. SDN is motivated by earlier network programmability efforts, including Active Networks [37], ForCES [38], RCP [39] and Tempest [40]. Unlike most earlier network programmability architectures, which explored clean-slate design of data plane protocols, SDN maintains backwards compatibility with existing network technologies. SDN design is driven by four major design goals: (i) network control and data plane separation; (ii) logical control centralization; (iii) open and flexible interfaces between control layers; and iv) network programmability.

SDN standardization efforts are primarily driven by the Open Network Foundation (ONF), while the IRTF SDNRG WG [41] explores complementary standards for the higher control layers. Similar standardization activities take place within various SDOs, namely the Broadband Forum (broadband network applications) and the International Telecommunication Union (ITU) study groups (SG) 11 (SDN signaling), SG 13 (SDN applications in future networks), SG 15 (transport network applications of SDN) and SG 17 (applications of SDN for secure services), but efforts in these SDOs are currently in early stages and provide initial problem statements and requirement analysis.

Fig. 3 presents an architectural model of an SDN control stack. The architecture separates the control functionalities into three distinct layers. The *data plane* is the bottom layer and contains all the network devices of the infrastructure. Data plane devices are designed to efficiently perform a restricted set of low-level traffic monitoring and packet manipulation functions and have limited control intelligence. Each devices implements one or more southbound Interfaces (SBIs) which enable control of the forwarding and resource allocation policy from external entities. SBIs can be categorized into control interfaces like OpenFlow [42] and PCE [43], designed to manipulate the device

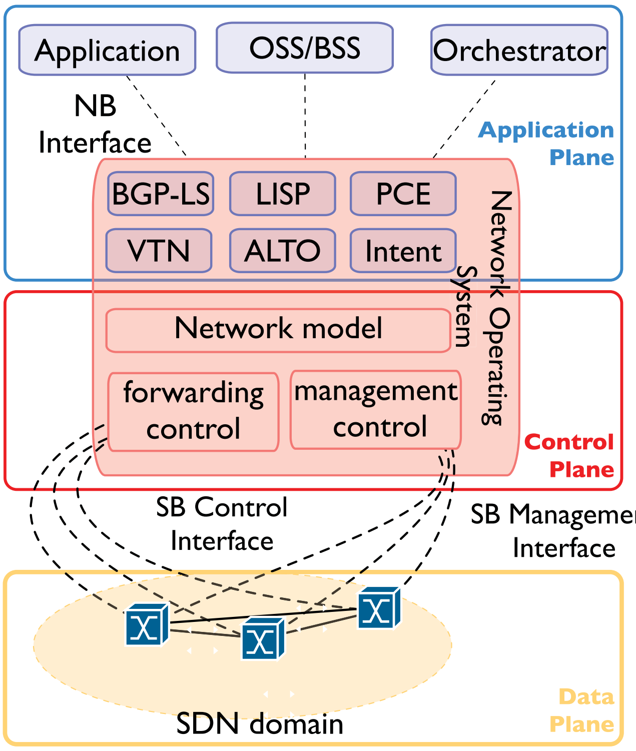

Fig. 3. The SDN architecture model can be separated in three layers: the data, control and application planes.

forwarding policy, and management interfaces, like NETCONF [44] and OF-CONFIG [45], designed to provide remote device configuration, monitoring and fault management. SDN functionality is not limited to networks supporting new clean-slate programmable interfaces and includes SBIs based on existing control protocol, like routing protocols.

The *control plane* is the middle layer of the architecture and contains the Network Operating System (NOS), a focal point of the architecture. A NOS aggregates and centralizes control of multiple data plane devices and synthesize new high-level Northbound Interfaces (NBIs) for management applications. For example, existing NOS implementations provide topology monitoring and resource virtualization services and enable high-level policy specification languages, among other functionalities. Furthermore, a NOS aggregates control policy requirements from management applications and provides them accurate network state information. The NOS is responsible to analyze policy requests from individual management applications, ensure conformance with the administrative domain policy, detect and mitigate policy conflicts between management applications and translate these requests into appropriate data plane device configurations. A key element for the scalability of the architecture is logical centralization of network control; a control plane can consist of multiple NOS instances, each controlling an overlapping network segment, and use synchronization mechanisms, typically termed as eastbound and westbound interfaces, to converge in a common network-wide view of the network state and policy between NOS instances. This way, an SDN control domain can recover from multiple NOS instance failures and the control load can be distributed across the remaining instances. Finally, the *application plane* is the top layer of the architecture and contains specialized applications that use NBIs to implement high-level NFs, like load balancing and resource management.

Detailed presentation of the standardization, research and implementation efforts in the SDN community are presented in [46]. For the rest of this section we focus on NBI standardization efforts. NBIs are crucial for service orchestration, since they enable control and monitoring of service connectivity and network resource utilization and flexible fault-management. Nonetheless, NBI standardization is limited and existing control interface and mechanism design is driven by NOS development efforts.

NBIs can be organized in two broad categories. The first category contains low-level information modeling NBIs. Information models converge the state representation of data plane devices and abstract the heterogeneity of SBIs. Network information models have been developed before the introduction of the SDN paradigm by multiple SDOs, like the ITU [47,48] and the Distributed Management Task Force (DMTF) [49]. Relevant to the SDN paradigm is the ONF information modeling working group (WG), which develops the Common Information Model (CoreModel) [50] specifications. The CoreModel is hierarchical and includes a core model, which provides a basic abstraction for data plane forwarding elements, and a technology forwarding and an application-specific model, which evolve the core model abstraction. CoreModel specifications exploit object inheritance and allow control applications to acquire abstract network connectivity information and, in parallel, access technology-specific attributes of individual network devices. The CoreModel adoption is limited and existing NOSes employ custom information models.

The second NBI category contains high-level and innovative control abstractions, exploring interfaces beyond the typical match-action-forward model. These interfaces are typically implemented as NOS management applications, use the information model to implement their control logic and are consumed by external entities, like the Operation Support System (OSS), the service orchestrator and other control applications. Effectively, these interfaces manifest the reference points between the Network and Service Orchestrator components (Fig. 1). For the rest of this section we elaborate on NBI formal specifications, as well as NBI designs developed in production NOSes.

We elaborate on legacy control interfaces implemented in SDN environment, as well as interfaces supported by the ONOS [51] and OpenDayLight (ODL) [52] projects, the most popular and mature open-source NOS implementations.

Path Computation. Path Computation Element (PCE) is a control technology which addresses resource and forwarding control limitations in label-switched technologies. Generalized Multi-Protocol Label Switching (GMPLS) and Multi-Protocol Label Switching (MPLS) technologies follow a distributed approach for path establishment. Switches use traffic engineering extensions to routing protocols, like OSPF-TE [53], to collect network resource and topology information. Path requests trigger a label switch to compute an end-to-end path to the destination network using its topology information and provisions the path using signaling protocols, like RSVP-TE [54]. A significant limitation in MPLS path computation is the increased computational requirements for the co-processor of edge label switches in large networks, while limited visibility between network layers or across administrative domains can lead to sub-optimal path selections. PCE proposes a centralized path computation architecture and defines a protocol which allows the network controller to receive path requests from the NMS and to configure paths across individual network forwarding elements. PCE control can be used by the service orchestrator to provision connectivity between the NF nodes.

The ONOS PCEP project[1] enables ONOS to serve Path Computation Client (PCC) requests and to manage label switched paths (LSP) and MPLS-TE tunnels. In addition, the PCEP project develops a path computation mechanism for the ONOS tunneling subsystem and provides tunnels as a system resource. Tunnel establishment support, both as L2 and L3 VPNs, is available to application through a RESTful NBI and applications are distinguished between tunnel providers and tunnel consumers.

LSP computation relies on network topology information, stored in a traffic engineering database (TED) and populated by an Interior Gateway Protocol (IGP). This information remains local within an Autonomous System (AS), limiting Path Computation in a single administrative domain. The IETF Inter-Domain Routing WG defines a mechanism to share link-state information across domains using the Network Layer Reachability Information (NLRI) field of the BGP protocol, standardized in the BGP-LS protocol extensions [55]. The ONOS BGP-LS project introduces support for the BGP-LS protocol (peering and link state information support) as SBI to complement the ONOS PCEP project 1.

The BGP-LS/PCEP module[2] of the ODL project implements support for the aforementioned protocols as a control application. Furthermore, the module supports additional PCE extensions, like stateful-PCE [56], PCEP for segment routing (Section 5.4), and secure transport for PCEP (PCEPS) [57]. Stateful-PCE introduces time, sequence and resource usage synchronization within and across PCEP sessions, allowing dynamic LSP management. Furthermore, PCEPS adds security extension to the control channel of the PCE protocol.

ALTO. The Application Layer Traffic Optimization [58] is an IETF WG developing specifications that allow end-user applications to access accurate network performance information. Distributed network applications, like peer-to-peer and content distribution, can improve their peer-selection logic using network path information towards alternative service end-points. This better-than-random decision improves the performance of bandwidth-intensive or latency-sensitive applications, while the network provider can improve link utilization across its network. The ALTO protocol enables a service orchestrator to monitor the network of the operator and make informed service deployment decisions. ODL provides an ALTO server module[2] with a RESTful ALTO NBI.

Virtual Tenant Networks. Virtual Tenant Networks (VTNs) [59] is a network virtualization architecture, developed by NEC. VTN develops an abstraction that logically disassociates the specification of virtual overlay networks from the topology of the underlying network infrastructure. Effectively, users can define any network topology and the VTN management system will map the requested topology over the physical topology. VTN enables seamless service deployment for the service orchestrator, by decoupling the deployment plan from the underlying infrastructure. The VTN abstraction is extensively supported by the ODL project.[2]

Locator/ID Separation. The IETF Locator/ID separation protocol (LISP) [60] is a network architecture addressing the scalability problems of routing systems at Internet-scale. LISP proposes a dual addressing mechanism, which decouples the location of a host from its unique identifier. LISP-aware end-hosts require only a unique destination end-point identifier (EID) to transmit a packet, while intermediate routing nodes use a distributed mapping service to translate EIDs to Routing Locations (RLOCs), an identifier of the network of the destination host. A packet is send to an Edge LISP router in the EID domain, where a LISP header with the RLOC address of the destination network is added. The packet is then routed across the underlay network to the destination EID domain. The LISP architecture provides a scalable mechanism for NFs connectivity and mobility.

ODL provides a LISP flow mapping module.[2] The module uses an SBI to acquire RLOC and EID information from the underlying network and exposes this information through a RESTCONF NBI. In addition, the NBI allows applications, like load balancers, to create custom overlay networks. The module is currently compatible with the Service Function Chain (SFC) (Section 5.3) functionality and holds future integration plans with group-based policy mechanisms.

Real time media. The ONF has currently a dedicated WG exploring standardization requirements for SDN NBIs. At the time of writing, the group has released an NBI specifications for a Real Time Media [61] control protocol, in collaboration with the International Multimedia Telecommunication Consortium (IMTC). The protocol allows end-user applications to communicate with the local network controller, discover available resources and assign individual flows to specific quality of experience (QoE) classes, through a RESTful API. ONF is currently developing a proof-of-concept implementation of the API as part of the ASPEN project [62].

Intent-based networking. Intent-based networking is a popular SDN NBI exploring the applicability of declarative policy languages in network management. Unlike traditional imperative policy language, Intent-based policies describe to the NOS the set of acceptable network states and leave low-level network configuration and adaptation to the NOS. As a result, Intents are invariant to network parameters like link outages and vendor variance, because they lack any implementation details. In addition, intents are portable across controllers, thus simplifying application integration and run-time complexity, but requires a common NBI across platforms, which is currently an active goal for multiple SDOs WG.

The IETF has adopted the NEMO specifications [63], an Intent-based networking policy language. NEMO is a Domain Specific Language (DSL), following the declarative programming paradigm. NEMO applications do not define the underlying mechanisms for data storage and manipulation, but rather describe their goals. The language defines three major abstractions: an `end-point`, describes a network end-point, a `connection`, describes connectivity requirements between network end-points, and an `operation`, describes packet operations. Huawei is currently leading an implementation initiative, based on ODL and the OPNFV project [64].

In parallel, the ONF has recently organized a WG to standardize a common Intent model. The group aims to fulfill two objectives: i) describe the architecture and requirements of Intent implementations across controllers and define portable intent expressions, and ii)

[1] https://wiki.onosproject.org/display/ONOS/Feature+Proposals
[2] https://wiki.opendaylight.org/view/Project_list

develop a community-approved information model which unifies Intent interfaces across controllers. The respective standard is coupled with the development of the Boulder framework [65], an open-source and portable Intent framework which can integrate with all major SDN NOSes. Boulder organizes intents through a grammar which consists of subjects, predicates and targets. The language can be extended to include constraints and conditions. The reference Boulder implementation has established compatibility with ODL through the Network Intent Composition (NIC) project, while ONOS support is currently under development.

Group-Based Policy (GBP) is an alternative Intent-based networking paradigm, developed by the ODL project. Based upon promise-theory [66], GBP separates application concerns and simplifies dependency mapping, thus allowing greater automation during the consolidation and deployment of multiple policy specifications. The GBP abstraction models policy using the notions of end-point and end-point groups and provides language primitives to control the communication between them. Developers can specify through GBP their application requirements and the relationship between different tiers of their application, while remaining opaque towards the topology and capabilities of the underlying network. The ODL GBD module provides an NBI[2] which leverages the low-level control of several network virtualization technologies, like OpenStack Neutron [67] and SFC (Section 5.3).

4.2. Application-based network operations (ABNO)

The evolution of the SDN paradigm has highlighted that clean-slate design approaches are prone to protocol and interface proliferation which can limit the evolvability and interoperability of a deployment. ABNO [68] s an alternative modular control architecture standard, published as an Area Director sponsored RFC document, and it reuse existing standards to provide connectivity services. ABNO by-design provides network orchestration capabilities for multi-technology and multi-domain environments, since it relies on production protocols developed and adopted to fulfill these requirements. The architecture enables network applications to automatically provision network paths and access network state information, controlled by an operator-defined network policy.

ABNO consists of eight functional blocks, presented in Fig. 4 along with their interfaces, but production deployments do not require to implement all the components. A core element of the architecture is the *ABNO controller*. The controller allows applications and NMS/OSS to specify end-to-end path requirements and access path state information. A path request triggers the controller to inspect the current network connectivity and resource allocations, and to provision a path which fulfills the resource requirements and does not violate the network policy. In addition, the controller is responsible to re-optimize

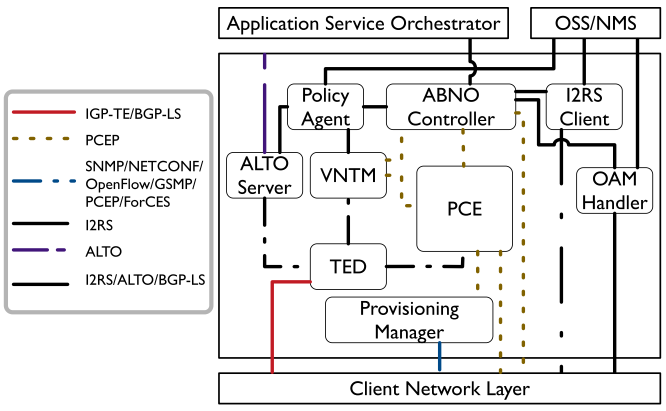

Fig. 4. The functional blocks of an ABNO architecture. Interface between functional block can re-use existing protocol standards.

paths at run-time, taking under consideration other path requests, routing state and network errors. The architecture contains an *OAM handler* to collect network error from all network layers. The OAM handler monitors the network and collects error notifications from network devices, using interfaces like IPFIX and NETCONF, which are correlated in order to synthesize high-level error reports for the ABNO controller and the NMS. In addition, the ABNO architecture integrates with the network routing policy through an *Interface to the Routing System (I2RS) client*. I2RS [69] is an IETF WG that develops an architecture for real-time and event-based application interaction with the routing system of network devices. Furthermore, the WG has developed a detailed information model [70] that allows external applications to monitor the RIB of a forwarding device. As a result, the I2RS client of the ABNO architecture aggregates information from network routers in order to adapt its routing policy, while it can by modify routing tables the routing policy to reflect path availability.

Path selection is provided by a *PCE controller*, while a *provisioning manager* is responsible for path deployment and configuration using existing control plane protocols, like OpenFlow and NETCONF. It is important to highlight that these functional blocks may be omitted in a production deployment and the architecture proposes multiple overlapping control channels. In addition, the architecture contains an optional *Virtual Network Topology Manager (VNTM)*, which can provision connectivity in the network physical layer, like configuring virtual links in WDM networks.

Topology discovery is a key requirement for the path selection algorithm of the PCE controller and the ABNO architecture uses multiple databases to store relevant information. The *Traffic-Engineering Database (TED)* is a required database for any ABNO architecture and contains the network topology along with link resource and capability information. The database is populated with information through traffic engineering extensions in the routing protocol. Optionally, the architecture suggests support for an *LSP* database, which stores information for established paths, and a database to store associative information between physical links and network paths, for link capacity prediction during virtual link provision over optical technologies.

A critical element for production deployment is the ability of the ABNO architecture to employ a common policy for all path selection decisions. The ABNO architecture incorporates a *Policy Agent* which is controlled by the NMS/OSS. The policy agent authenticates requests, maintains accounting information and reflects policy restrictions for the path selection algorithm. The policy agent is a focal point in the architecture and any decision by the ABNO controller, the PCE controller and the ALTO server requires a check with the active network policy.

In addition to the ABNO control interfaces, the architecture provides additional application interfaces which expose network state information through an *ALTO server*. The server uses the ALTO protocol to provide accurate path capacity and load information to applications and assist the application server selection process and performance monitoring.

A number of ABNO-based implementations exist detailing how the architecture was used to orchestrate resources in complex network environments, including: iONE [71] for content distribution in the telecom Cloud [72], and Adaptive Network Manager [73] for co-ordinating operations in flex-grid optical and packet networks [74]. The large telecom vendor Infinera and network operator Telefonica, also provided a joint demonstration to orchestrate and provision bandwidth services in real-time (”Network as a Service - NaaS”) across a multi-vendor IP/MPLS and optical transport network, using a variety of APIs [75].

5. Function orchestration standardization

Along with the ability to control end-to-end connectivity, service

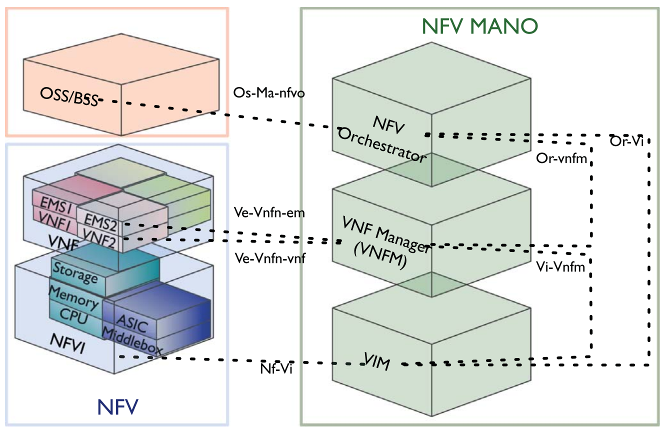

Fig. 5. ETSI NFV management and orchestration architecture.

orchestration requires support for automated control, management and configuration of NFs. Currently, NFs appear as a bump on the wire. In addition, NF implementations rely on specialized devices, while their control and management interfaces exhibit significant proliferation and heterogeneity and are not integrated with the network control plane. As a result, service deployment requires extensive human intervention to populate the network forwarding policy with static configurations that steer traffic to the desired NFs, resulting in limited service agility constrained by the underlying network topology. These limitations convolute the management of network services and increase service lead-times, especially for highly available services. Service management is further convoluted by the introduction of virtualized and software-based NFs (VNFs). Although VNFs provide service flexibility and elasticity, they introduce new functional properties, like lower performance predictability and reliability. Mixing VNF with traditional single-purpose NFs, must take under consideration these characteristics and requires fine-grain dynamic traffic steering mechanisms to ensure service liveness.

To address challenges towards flexible and agile services, multiple standardization bodies have proposed architectures, protocols, and control interfaces which enable seamless and dynamic function management. This section presents some popular NFV standardization efforts, namely the ETSI NFV Management and Orchestration (MANO) specifications (Section 5.1), the Metro Ethernet Forum (MEF) Lifecycle Service Orchestration (LSO) (Section 5.2) architecture, exploring the management organization of NFV solutions, and the IETF *Service Function Chain (SFC)* (Section 5.3) and *Segment Routing (SR)* (Section 5.4), designed to simplify the translation of service connectivity requirements into network policy.

5.1. NFV management and orchestration (NFV MANO)

The ETSI is the first SDOs to explore the applicability of the NFV paradigm in operator infrastructures [26] and to develop Proof of Concept [76] NFV implementations. Furthermore, ETSI leads the design of the popular NFV MANO architecture [77]. NFV standardization is not limited to ETSI, and other standardization bodies, like the IETF NFVRG charter [78], the Open Platform for NFV (OPNFV) industrial forum [64] and the TM Forum's ZOOM,[3] develop MANO reference implementations and propose extensions to the MANO architecture.

The MANO specifications abstract the control of virtualized infrastructures and VNF instances to external entities, like the OSS/BSS and the service orchestrator of an operator. It is currently the most popular NFV management framework, with numerous open-source and commercial implementations. Operators explore the adoption of MANO-compatible managements systems for various compounding reasons. Firstly, NFV MANO is a flexible component-based architecture which re-uses existing infrastructure management frameworks, like SDN NOSes and the OpenStack framework. Therefore, existing components can be extended by vendors, simplifying the development of NFV platforms. Secondly, the maturity and relatively detailed specification of the MANO components enable seamless interoperability between implementations from different vendors. Thirdly, the architecture provides by-design multiple carrier-grade features, like scalable hierarchical control, billing, and flexible service and function lifecycle specification.

Integration between the different functional components of the ETSI architecture is achieved through reference points, a distributed information plane which models state updates and control operations. The root element of the information plane is the Network Service (NS), which represents the service chain of a service. A NS consists of one or more *Virtual Network Functions (VNF)*, like firewalls or load balancers, connected using *Virtual Links*, while a *VNF Forwarding Graph (VNFFG)* defines VNF ordering. Furthermore, a NS may include *Physical Network Functions (PNF)*, available in the underlying network infrastructure. Finally, the MANO information model defines data repositories of NS templates, VNF catalogues, and NFVI resources, which simplify the specification and deployment of a NS.

For the rest of this section, we elaborate on the design of the MANO architecture and identify some design limitations. Fig. 5 depicts a diagram of the MANO components with the left-hand side representing the infrastructure and the right-hand side representing the management of the infrastructure. The architecture separates VNF management into three distinct layers, in an effort to support by-design clean control separation between the hosting infrastructures and the NFV managers.

Virtualized Infrastructure Manager (VIM). The VIM provides direct control and monitoring capabilities for a single NFV Infrastructure (NFVI) domain to the upper layers of the MANO architecture. VIM responsibilities include the management of the compute, network, and storage resources of a datacenter and it exposes interfaces for resource control and VNF image management. Current implementations re-use existing Cloud Management Systems (CMS), like the popular and open-source OpenStack, to realize the VIM layer. Nonetheless, the design goals of existing CMSs cannot accommodate some VIM requirements, like carrier-grade support, high-performance I/O and fine-grain and timely resource control [79,80]. Currently, OPNFV, in collaboration with ETSI, designs and develops new open-source VIM and infrastructure virtualization platforms, that bridge this requirement gap.

Virtual Network Function Manager (VNFM). The VNFM sits between the NFVO and the VIM systems and is responsible for the lifecycle management of individual VNF instances, including VNF configuration, monitoring, termination, and scaling. VNF management is typically realized using an *Element Manager (EMS)* which monitors and reports the state of each VNF to the VNFM and is capable to modify the configuration of the VNF. The deployment of an NFVM is not mandatory according to the MANO specifications and the functionality of this layer can be implemented by the NFV orchestrator. Current MANO frameworks either lack an NFVM or develop a very thin adaptation layer between the NFV orchestrator and the VIM, responsible to propagate VNF image deployment requests. Nonetheless, a VNFM can enable seamless interoperability between VNF implementations from different vendors and across cloud infrastructures [81].

Network Functions Virtualization Orchestrator (NFVO). The NFVO is responsible for the deployment and dynamic re-optimization of network services. Effectively, the NFVO receives NS requests from external entities, like the OSS and the service orchestrator, and coordinates the deployment and configuration of VNF instances across the NFVI domains. In parallel, the NFVO monitor the service perfor-

[3] https://www.tmforum.org/zoom/

mance and dynamically re-optimizes the deployment of VNF instance to meet the NS requirements. When creating a new NS, the NFVO optimizes placement of VNFs whilst ensuring sufficient resources and connectivity are available. Current NFVO implementations provide a thin layer capable to launch and destroy VNF chains across the NFVI domains of the operator and provide limited support for dynamic re-optimization of the service deployment.

5.2. *MEF lifecycle service orchestration (LSO)*

The MEF is an industrial forum, responsible for the standardization of Carrier Ethernet (CE) technologies. Furthermore, it steers the standardization efforts for the MEF LSO [82], an architecture aiming to improve automation in network service management. MEF extends the MANO architecture and introduces support for end-to-end network infrastructure management, capitalizing on the flexible control of CE technologies. LSO targets challenges of delivering Network as a Service (NaaS) functionalities in the operator infrastructure, such as on-demand, agility, and heterogeneity of virtual and physical NFs. LSO refines the service lifecycle model of the MANO standards and introduce new lifecycle capabilities, including mechanisms to automate network service request *fulfillment*, *control* of service resource and scaling, enhanced *performance* monitor and guarantees and *assurances* for service survivability. LSO aims to improve the time to establish and modify services for their future Internet vision [82]. The development of the LSO standards is still in early stages and it currently focuses on service requirement specification in order to drive the architecture design.

5.3. *Service function chain (SFC)*

SFC is a recently formed IETF WG which aims to define the architectural principles and protocols for the deployment and management of NF forwarding graphs. An SFC deployment operates as a network overlay, logically separating the control plane of the service from the control of the underlying network. The overlay functionality is implemented by specialized forwarding elements, using a new network header. Fig. 6 presents an example deployment scenario of an SFC domain.

An administrative network domain can contain one or more *SFC domains*. An SFC domain is a set of SFC-enabled network devices sharing a common information context. The information context contains state regarding the deployed service graphs, the available paths for each service graph and classification information mapping incoming traffic to a service path. An SFC-specific header is appended on all packets on the edges of the SFC domain by an *SFC-Classifier*. The SFC-Classifier assigns incoming traffic to a service path by appending an appropriate SFC header to each packet. For outgoing traffic, the SFC-Classifier is responsible to remove any SFC headers and forward each packet appropriately. Once the packet is within the SFC domain, it is forwarded by the classifier to an *SF Forwarder (SFF)*, an element responsible to forward traffic to an SF according to the service function ordering. Finally, the architecture is designed to accommodate both SFC-aware and legacy NFs. The main difference between them is that the SFC-aware NFs can parse and manipulate SFC headers. For legacy NFs, the architecture defines a specialized element to manipulate SFC headers on behalf of the service function, the *SFC-Proxy*. The network overlay of the SFC architecture is realized through a new protocol layer, the Network Service Header (NSH) [83]. NSH contains information which define the position of a packet in the service path, using a service path and path index identifiers, and carry metadata between service functions regarding policy and post-service delivery.

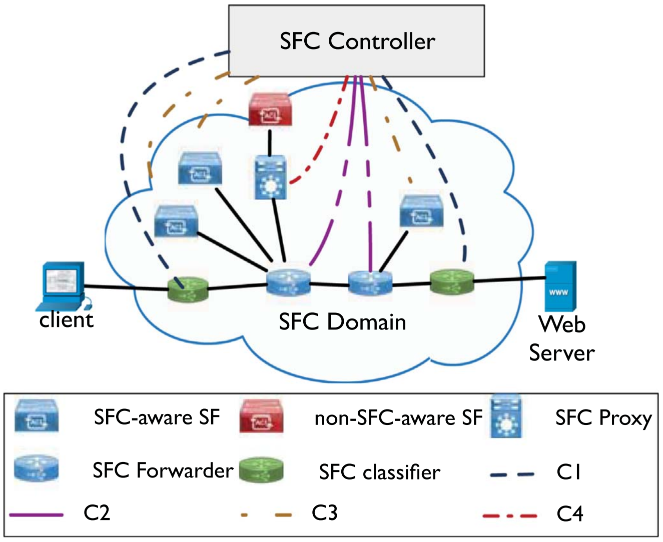

Fig. 6. IETF SFC architecture.

Highly relevant for service orchestration is the control and management interfaces of the SFC architecture. At the time of writing, the SFC WG currently explores the SFC control channel requirements and initial design goals [84] define four main control interfaces. *C1* is the control channel of the SFC-Classifier and allows manipulation of the classification policy which assigns incoming traffic to specific service paths. This control interface can be used to load balance traffic between service paths and optimize resource utilization. *C2* is a control channel of the SFF forwarding policy and exposes monitoring information, like latency and load. *C3* is the control protocol used to aggregate status, liveness and performance information from each NF-aware service function. Finally, the controller can use the *C4* protocol to configure SFC-Proxies with respect to NSH header manipulation before and after a packet traverses an SFC-unaware NF. In parallel, the WG has proposed a set of YANG models to implement the proposed control interfaces [85]. Furthermore, the WG has also specified a set of YANG models for the management interface of an SFC controller [84]. This interface provides information about the liveness of individual SFC paths, topological information for the underlying SFC infrastructure, performance counters and control of the fault and error management strategies. In addition, the management interface allows external applications to re-optimize service paths and control load balancing policy.

At the time of writing, multiple open-source platforms introduce SFC support. The Open vSwitch soft-switch has introduced SFC support both in the data and the control (OpenFlow extensions) plane. The OpenStack cloud management platform exploits the Open vSwitch SFC support and implements a high-level SFC control interface [86]. Furthermore, the ONOS controller currently supports SFC functionality using VTN overlays, while ODL implements SFC support using LISP tunnels. In addition, ONF has released recommendations for an L4-L7 SFC architecture [87] which uses OpenFlow as the SBI of the SFC controller and explores the applicability and required extension to the OpenFlow abstraction to improve support for SFF elements.

5.4. *Segment routing (SR)*

Segment Routing (SR) [88] is an architecture for the instantiation of service graphs over a network infrastructure using source routing mechanisms, standardized by the IETF Source Packet Routing in Networking (SPRING) WG [89].

SR is a data plane technology and uses existing protocols to store instructions (segments) for the packet path in its header. SR segments can have local or global semantics, and the architecture defines three segments types: a node segment forwards a packet over the shortest path towards a network node, an adjacency segment forwards the packet through a specific router port and a service segment introduces service differentiation on a service path. Currently, the SR architecture has defined a set of extensions for the IPv6 [90] and the MPLS [91] protocols, which define protocol-compliant mechanisms to store the segment stack and the active segment pointer in the protocol header. In addition, to enable dynamic adaptation of the forwarding policy, the architecture defines a set of control operations for forwarding elements to manipulate the packet segment list and to update established paths dynamically.

The selection of the packet path is implemented on the edge routers of the SR domain. The architecture specifies multiple path selection mechanisms, including static configurations, distributed shortest-path selection algorithms and programmatic control of segment path using SDN SBIs. The network IGP protocol can be used to provide segment visibility between routers and a YANG management interface is defined for SR segment information retrieval and SR routing entry control.

SR provides a readily-available framework to instantiate service forwarding graphs. A forwarding graph can be implemented as a segment stack and existing VNFs can be integrated with the architecture by introducing appropriate support for MPLS and IPv6 SR extensions. In comparison to the SFC architecture, SR provides a simpler architecture which does not require deployment of new network elements. Nonetheless, SFC provides wider protocol support and the architecture is designed to support different data plane technologies, while SR is closely aligned with MPLS technologies.

SR support is currently introduced in both major SDN NOSes. The ONOS project has introduced support for SR to implement CORD, a flexible central office architecture designed to simplify network service management [92]. Similarly, ODL supports SR functionality using MPLS labels and the PCE SBI module. In parallel, CISCO has introduced SR support in recent XR IOS versions [93].

6. Challenges and future directions

A variety of industry challenges remain for the standardization of key orchestration technologies. Some of the protocol solutions discussed in this paper are immature and will require further investigation and development before they can be operationalized and used by operators. In some cases, new forwarding mechanisms lack sufficient security and operational considerations required for complex and large-scale environments. The rest of this section outlines areas of new research and standardization efforts and their importance for network service orchestration.

6.1. In-operation analysis and network telemetry

he increasing demand for dynamic resource, function and connectivity provision in an orchestrated infrastructure can increase network incidents and unregulated network changes. The success of a service orchestrator depends on its ability to measure the network performance, to assess service quality using a small set of metrics and to provide network diagnosis and root cause analysis during service disruptions. In parallel, the orchestrator must support network resource scheduling which can adapt to near real-time service demands (“in-operation”) [94].

To investigate network problems or identify the severity of major network events or interruptions, a network health index or network key performance index (KPI) or key quality index (KQI) is required. Generating the KPI or KQI would require data collection from various data sources using a set of automated communication processes and transmit them to one or more data aggregation services. This process is known as *network telemetry*.

The data collected from data sources include network performance data, network logging data, network warning and defects data, network statistics and state data, and network resource operation data (*e.g.*, operations on RIBs and FIBs). The process and ability to normalize the data to derive several end-to-end network composite metrics that reflect the network performance and quality from different perspectives, like network diagnosis, network performance, network QoS, network security. These end-to-end metrics can then be used for in-operation planning.

6.2. Orchestrator scalability

The size and scale of service orchestration interfaces manifest a complex distributed computing system. Operator infrastructures contain multiple computational resources (*i.e.*, CPU, memory, storage, and function) that are connected via the network and together they perform a task. Logical centralization for the infrastructure control and management systems, where a group of control elements exposes a unified and centralized abstraction to the layer above, has become a key design goal.

The CAP theorem [95] identifies three characteristics that are universally desirable, but cannot be met concurrently by any distributed system: *Consistency*, describes the ability of the system to respond identically to a request no matter which element receives the request; *Availability*, describes the ability of the system to always respond to a request; and *Partition Tolerance*, describes the ability of the system to function uninterrupted when nodes or communications links fail.

An orchestrator will act on request and connect to the various control elements. Tolerance to loss of connectivity from the orchestrator and various controllers is typically not discussed by most of the technologies discussed in this survey paper. The consistency, availability and partitioning issues may be solved by clustering critical components and duplicating databases, but large-scale resource pooling and state synchronization challenges will need to be addressed in the protocol and architecture design phase. It is critical for SDO to understand the consistency, performance and resilience requirements of each orchestration interface and define operational semantics for control operation.

6.3. Security and trust

The traditional attack vectors on traffic flows, switches, and functions, and recovery and fault diagnosis, have resulted in new security issues that are specific to SDN and NFV [96,97]. The features, capabilities and services outlined in our survey will introduce faults and risks that expose network infrastructure to threats that did not previously exist, or were ring-fenced by single OSS platforms, and are significantly more serious, with a greater potential for harm. Furthermore, security flaws can result when an open source project has a weak security focus (often the result of critical technology with too few reviewers and maintainers). This result has manifested recently in OpenSSL (HeartBleed), and is now being addressed through the Linux Foundation critical infrastructure project (for OpenSSL, OpenSSH and NTPd).

In co-operative controller environments or orchestrators that are capable of directly accessing and manipulating another technology or administrative domain controller, the risks associated with one compromised entity are now compounded, as attackers are able to attack a single resource control point. This is distinct from a larger number of autonomous assets in a completely distributed control architecture. Automation via orchestration is a double-edged sword; it offers flexibility to implement new, innovative and market-driven applications but it also opens the door to malicious and vulnerable applications. A sufficient *Trust Model* must be developed for SDN-based and NFV-based infrastructures, implementing robust authentication and enforcing different authorization levels during application registration to the orchestrator, in order to limit the exposure to misconfiguration, and malicious intent.

6.4. Service modeling

An important step towards effective network services orchestration is the development of models which capture the resource requirements, configuration parameters, performance metrics and fault management of network services. These models can drive the development of the interfaces between applications, service consumers and the service orchestrator. Standardizing a common set of service models can enable orchestrator-application interoperability between operators and address limitations arising in the deployment of services that span across multiple administrative domains.

Efforts towards service modeling are fairly recent and their outcomes are still limited. We identify two relevant SDO efforts: the Topology and Orchestration Specification for Cloud Applications (TOSCA) from the Organization for the Advancement of Structured Information Standards (OASIS) and the IETF NETCONF Data Modeling Language (NETMOD) WG. The TOSCA technical committee (TC) recently expanded its scope with a new goal to model VNF network services. At the time of writing, the TC has released a draft model [98], closely aligned with the information points in the ETSI MANO architecture. The IETF NETMOD WG provides a richer portfolio of model specifications, developed using the YANG [99] data modeling language. The respective models can be classified in two broad categories: network element models and network service models [100]. Relevant to network service modeling are the latter models, but the scope of these models remains limited and primarily focuses on connectivity services.

One of the key challenges towards network service modeling, is the definition of unified configuration and management VNF interfaces. Effectively, the interface between the VNF EMS layer and the VNFM service currently lacks standardization. VNF appliances comes in many different shapes and sizes and operate across all network layer. The high dimensionality of VNF interfaces can significantly impair automation in service orchestration. Relevant efforts in cloud computing have deliver frameworks, like Ansible [101] and Chef [102], which simplify the deployment of web services for large scale systems using configuration template. These systems provide *cookbooks* containing service *recipes* which abstract and automate web service and VM configuration. These approaches should be revisited and adapted in the context of network service deployment and configuration practices.

7. Summary

Operators currently face significant challenges to maintain profitability over their infrastructures and, in parallel, support network service innovation. Modern network infrastructures are complex systems, comprising of heterogeneous technologies, each with different proprietary configuration and management interfaces. Given the relatively long deployment times and static nature of existing customer services, the network service deployment and management is achieved using limited cross function collaboration, system focused and top-down command and control.

A key goal for operators is the development of new network service orchestration mechanisms which provide convergence between network technologies, automation in the deployment and management of network service and flexible and cross-layer resource control and provision. Towards this goal, new technological paradigms, including SDN and NFV, and new network architectures, such as SFC and SR, provide the opportunity to augment elasticity, programmability, interoperability and agility in the control and management of operator infrastructures and reduce CAPEX and OPEX.

This paper surveyed the standardization activities carried out in the recent years in the context of network service orchestration, in an effort to aid researchers and practitioners to understand the capabilities of the relevant technologies. We presented a simple architectural model for network service orchestration and we identified two principal elements in the management and control of operator infrastructures: network and NF orchestration. For each element, we presented the predominant architectural specifications and elaborated on the interfaces that each technology provides. Finally, we examined a number of future directions for the relevant SDO.

Acknowledgements

The authors are grateful to the UK Engineering and Physical Sciences Research Council (EPSRC) for funding the TOUCAN project (EP/L020009/1), which supported much of the work presented in this paper; and to the EPSRC , UK and BT for funding Daniel King's Industrial CASE Ph.D. research.

References

[1] R. Chua, SDXcentral: Christos Kolias, Orange Kicks off NFV Interview Series, 2013 URL ⟨https://www.sdxcentral.com/articles/interview/christos-kolias-sdncentrals-nfv-interview-series/2013/07/⟩.
[2] ITU, ICT Facts and Figures: The World in 2015, May 2015 ⟨http://www.itu.int/en/ITU-D/Statistics/Documents/facts/ICTFactsFigures2015.pdf⟩.
[3] Cisco, Cisco Visual Networking Index: Forecast and Methodology, 2014–2019 White Paper.
[4] Alcatel-Lucent, The Declining Profitability Trend of Mobile Data: What Can be Done? ⟨http://www3.alcatel-lucent.com/belllabs/advisory-services/documents/Declining_Profitability_Trend_of_Mobile_Data_EN_Market_Analysis.pdf⟩.
[5] D. Ward, Open standards, open source, open loop, IETF J. 10 (2015).
[6] A. Aguado, M. Davis, S. Peng, M.V. Ãlvarez, V. LÃpez, T. Szyrkowiec, A. Autenrieth, R. Vilalta, A. Mayoral, R. Muoz, R. Casellas, R. Martnez, N. Yoshikane, T. Tsuritani, R. Nejabati, D. Simeonidou, Dynamic virtual network reconfiguration over sdn orchestrated multitechnology optical transport domains, J. Lightwave Technol. 34 (8) (2016).
[7] A. Csaszar, W. John, M. Kind, C. Meirosu, G. Pongracz, D. Staessens, A. Takacs, F. J. Westphal, Unifying cloud and carrier network: EU FP7 Project UNIFY, in: IEEE/ACM UCC, Dec 2013.
[8] T. Benson, A. Akella, D. Maltz, Unraveling the complexity of network management, in: NSDI. USENIX, 2009.
[9] L. Lei, SDN orchestration for dynamic end-to-end control of data center multi-domain optical networking, China Commun. 12 (8) (2015).
[10] Y. Yoshida, A. Maruta, K. Kitayama, M. Nishihara, T. Tanaka, T. Takahara, J.C. Rasmussen, N. Yoshikane, T. Tsuritani, I. Morita, S. Yan, Y. Shu, M. Channegowda, Y. Yan, B.R. Rofoee, E. Hugues-Salas, G. Saridis, G. Zervas, R. Nejabati, D. Simeonidou, R. Vilalta, R. Munoz, R. Casellas, R. Martinez, M. Svaluto, J.M. Fabrega, A. Aguado, V. Lopez, J. Marhuenda, O.G. de Dios, J.P. Fernandez-Palacios, First international SDN-based network orchestration of variable-capacity OPS over programmable flexi-grid EON, in: OFC, March 2014.
[11] S. Sahhaf, W. Tavernier, J. Czentye, B. Sonkoly, P. Skoldstrom, D. Jocha, J. Garay, Scalable architecture for service function chain orchestration, in: EWSDN, Sept 2015.
[12] J. Ellerton, A. Lord, P. Gunning, K. Farrow, P. Wright, D. King, D. Hutchison, Prospects for software defined networking and network function virtualization in media and broadcast, in: SMPTE, Oct 2015.
[13] B. Ethirajulu, Above & Beyond Mano,⟨https://www.ericsson.com/spotlight/cloud/blog/2015/12/02/beyond-mano/⟩.
[14] China Mobile, C-RAN: The Road Towards Green RAN, 2011.
[15] S. Bhaumik, S.P. Chandrabose, M.K. Jataprolu, G. Kumar, A. Muralidhar, P. Polakos, V. Srinivasan, T. Woo, CloudIQ: A framework for processing base stations in a data center, in: Mobicom. ACM, 2012.
[16] N. Nikaein, M.K. Marina, S. Manickam, A. Dawson, R. Knopp, C. Bonnet, OpenAirInterface: a flexible platform for 5G research, SIGCOMM Comput. Commun. Rev. 44 (5) (. 2014).
[17] Alcater-Lucent, vRAN, URL ⟨https://www.alcatel-lucent.com/solutions/vran⟩.
[18] R. Riggio, K. Gomez, L. Goratti, R. Fedrizzi, T. Rasheed, V-Cell: Going beyond the cell abstraction in 5G mobile networks, in: IEEE NOMS, May 2014.
[19] Common Public Radio Interface (CPRI); Interface Specification V6.0 (2013-08-30), 2013 URL ⟨http://www.cpri.info/downloads/CPRI_v_6_0_2013-08-30.pdf⟩.
[20] 3G PPP, TR 32.842: Telecommunication management; Study on network management of virtualized networks, 2015 URL ⟨https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?SpecificationId=2248⟩.
[21] 5G PPP, 5G Vision: The 5G Infrastructure Public Private Partnership: The Next Generation of Communication Networks and Services, 2015 URL ⟨https://5g-ppp.eu/wp-content/uploads/2015/02/5G-Vision-Brochure-v1.pdf⟩.
[22] Ericsson, 5G systems: Enabling The Transformation of Industry And Society, 2017. ⟨https://www.ericsson.com/assets/local/publications/white-papers/wp-5g-systems.pdf⟩.
[23] I.F. Akyildiz, P. Wang, S.-C. Lin, SoftAir: a software defined networking architecture for 5G wireless systems, Comput. Netw. 85 (2015).
[24] Open5GCore: The Next Mobile Core Network Testbed Platform, ⟨http://www.open5gcore.org⟩.
[25] Alcatel-Lucent, Rapport cloud communications, ⟨https://www.alcatel-lucent.com/solutions/cloud-communications⟩.
[26] ETSI, Network Functions Virtualisation (NFV); Use Cases, 2013.
[27] B. Krishnamurthy, C. Wills, Y. Zhang, On the use and performance of content distribution networks, in: IMW. ACM, 2001.

[28] Akamai, Akamai Facts and Figures, 2015 ⟨https://www.akamai.com/us/en/about/facts-figures.jsp⟩.
[29] Netflix, Netflix Peering Locations, 2006 ⟨https://openconnect.netflix.com/en/peering-locations/⟩.
[30] X. Wang, M. Chen, T. Taleb, A. Ksentini, V.C.M. Leung, Cache in the air: exploiting content caching and delivery techniques for 5G systems, IEEE Commun. Mag. 52 (2) (2014).
[31] X. Li, K. Kanonakis, N. Cvijetic, A. Tanaka, C. Qiao, T. Wang, Joint bandwidth provisioning and cache management for video distribution in software-defined passive optical networks, in: OFC. OSA, 2014.
[32] P. Georgopoulos, M. Broadbent, A. Farshad, B. Plattner, N. Race, Using software defined networking to enhance the delivery of video-on-demand, Comput. Commun. 69 (2015) 9.
[33] B. Frank, I. Poese, Y. Lin, G. Smaragdakis, A. Feldmann, B. Maggs, J. Rake, S. Uhlig, R. Weber, Pushing CDN-ISP collaboration to the limit, SIGCOMM Comput. Commun. Rev. 43 (3) (2013).
[34] R.D. Doverspike, K.K. Ramakrishnan, and C. Chase, Guide to reliable internet services and applications. London: Springer, 2010, ch. Structural Overview of ISP Networks, pp. 19–93.
[35] British Telecom, BTnet: Market Leading Leased Line Internet, Aug. 2015. ⟨http://business.bt.com/assets/pdf/broadband-and-internet/datasheet/BTnet_target_availability.pdf⟩.
[36] Open Network Foundation, Software-Defined Networking: The New Norm for Networks, 2012 ⟨https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf⟩.
[37] D.L. Tennenhouse, J.M. Smith, W.D. Sincoskie, D.J. Wetherall, G.J. Minden, A survey of active network research, IEEE Commun. Mag. 35 (1) (1997).
[38] A. Doria, J.H. Salim, R. Haas, H. Khosravi, W. Wang, L. Dong, R. Gopal, J. Halpern, Forwarding and control element separation (ForCES) protocol specification, Internet RFC, RFC 5810 (2010).
[39] M. Caesar, D. Caldwell, N. Feamster, J. Rexford, A. Shaikh, J. van der Merwe, Design and implementation of a routing control platform, in: NSDI. USENIX, 2005.
[40] J.E. van der Merwe, S. Rooney, L. Leslie, S. Crosby, The Tempest-a practical framework for network programmability, IEEE Network 12 (May (3)) (1998).
[41] IRTF, IRTF Software-Defined Networking Research Group, 2015 URL ⟨https://irtf.org/sdnrg⟩.
[42] Open Network Foundation, OpenFlow Switch Specifications 1.5.0, 2015 URL ⟨https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-switch-v1.5.1.pdf⟩.
[43] J. Vasseur, J.L. Roux, Path computation element (PCE) communication protocol (PCEP), Internet RFC, RFC 5440, 2009.
[44] R. Enns M. Bjorklund J. Schoenwaelder A. Bierman, Network configuration protocol (NETCONF), Internet RFC, RFC 6241, 2011.
[45] Open Network Foundation, OF-CONFIG 1.2: OpenFlow Management and Configuration Protocol, 2014 URL ⟨https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow-config/of-config-1.2.pdf⟩.
[46] D. Kreutz, F.M.V. Ramos, P.E. Veríssimo, C.E. Rothenberg, S. Azodolmolky, S. Uhlig, Software-defined networking: a comprehensive survey, Proc. IEEE 103 (1) (2015).
[47] ITU, ITU-T Recommendation M.3100: Generic Network Information Model, 2005 URL ⟨https://www.itu.int/rec/T-REC-M.3100/en⟩.
[48] ITU, ITU-T Recommendation M.3102: Unified Generic Management Information Model for Connection-oriented and Connectionless Networks,⟨https://www.itu.int/rec/T-REC-M.3102/en⟩, 2011.
[49] DMTF, Dmtf Common Information Model,⟨http://www.dmtf.org/standards/cim⟩, 2016.
[50] Open Network Foundation, Core Information Model (CoreModel),⟨https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/ONF-CIM_Core_Model_base_document_1.1.pdf⟩, 2015.
[51] P. Berde, M. Gerola, J. Hart, Y. Higuchi, M. Kobayashi, T. Koide, B. Lantz, B. O'Connor, P. Radoslavov, W. Snow, G. Parulkar, ONOS: Towards an Open, Distributed SDN OS, in: HotSDN. ACM, 2014.
[52] The OpenDaylight Platform, ⟨https://www.opendaylight.org/⟩.
[53] D. Katz K. Kompella D. Yeung, Traffic engineering (TE) extensions to OSPF version 2, Internet RFC, RFC 3630, September 2003.
[54] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, G. Swallow, RSVP-TE: extensions to RSVP for LSP tunnels, Internet RFC, RFC 3209, December 2001.
[55] J. Medved, S. Previdi, S. Ray, H. Gredler, A. Farrel, North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP, RFC 7752, Mar. 2016. [Online]. Available:⟨https://rfc-editor.org/rfc/rfc7752.txt⟩.
[56] J. Medved, I. Minei, E. Crabbe, R. Varga, PCEP Extensions for Stateful PCE, Internet Engineering Task Force, Internet-Draft draft-ietf-pce-stateful-pce-14, Mar. 2016, work in Progress. [Online]. Available: ⟨https://tools.ietf.org/html/draft-ietf-pce-stateful-pce-14⟩.
[57] D. Lopez, O.G. de Dios, Q. Wu, D. Dhody, Secure Transport for PCEP, Internet Engineering Task Force, Internet-Draft draft-ietf-pce-pceps-10, Jul. 2016, work in Progress. [Online]. Available: ⟨https://tools.ietf.org/html/draft-ietf-pce-pceps-10⟩.
[58] J. Seedorf, E. Burger, Application-Layer Traffic Optimization (ALTO) Problem Statement, RFC 5693, Oct. 2015.
[59] NEC, ProgrammableFlow: Redefining Cloud Network Virtualization With OpenFlow, 2011.
[60] D. Farinacci, V. Fuller, D. Meyer, D. Lewis, The Locator/ID Separation Protocol (LISP), RFC 6830, Oct. 2015. [Online]. Available:⟨https://rfc-editor.org/rfc/rfc6830.txt⟩.
[61] Open Network Foundation, ONF TR-517: Real Time Media NBI REST Specification, ⟨http://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/Real_Time_Media_NBI_REST_Specification.pdf⟩, 2015.
[62] Open Networking Foundation, Project ASPEN: real time media interface specification, ⟨http://opensourcesdn.org/projects/project-aspen-real-time-media-interface-specification/⟩.
[63] S. Hares, Intent-Based Nemo Overview, Internet Draft, RFC, Oct. 2015. [Online]. Available:⟨https://tools.ietf.org/pdf/draft-hares-ibnemo-overview-01.pdf⟩.
[64] C. Price, S. Rivera, OPNFV: An open platform to accelerate NFV,⟨https://networkbuilders.intel.com/docs/OPNFV_WhitePaper_Final.pdf⟩, 2012.
[65] Open Networking Foundation, Project BOULDER: intent northbound interface (NBI),⟨http://opensourcesdn.org/projects/project-boulder-intent-northbound-interface-nbi/⟩.
[66] P. Borril, M. Burgess, T. Craw, M. Dvorkin, A promise theory perspective on data networks, arXiv preprint arXiv:1405.2627, 2014.
[67] OpenStack, Neutron Developer Documentation,⟨http://docs.openstack.org/developer/neutron/⟩.
[68] D. King, A. Farrel, A PCE-Based architecture for application-based network operations, Internet RFC, RFC 7491, March 2015.
[69] A. Atlas, T. Nadeau, D. Ward, Problem statement for the interface to the routing system, Internet RFC, RFC 7920, June 2016.
[70] J. Clarke, G. Salgueiro, C. Pignataro, Interface to the Routing System (I2RS), Traceability: Framework and Information Model Internet RFC, RFC 7922, June 2016.
[71] L. Gifre, N. Navarro, A. Asensio, M. Ruiz, L. Velasco, iONE: An environment for experimentally assessing in-operation network planning algorithms, in: ICTON, July 2015.
[72] LluÃs Gifre, Massimo Tornatore, Luis M. Contreras, Biswanath Mukherjee, Luis Velasco, ABNO-driven content distribution in the telecom cloud, Opt. Switch. Netw. (2015).
[73] A. Aguado, V. Lopez, J. Marhuenda, O.G. de Dios, J.P. Fernandez-palacios, ABNO: a feasible SDN approach for multivendor IP and optical networks [Invited], IEEE/OSA J. Opt. Commun. Netw. 7 (2) (2015).
[74] V. Lopez, O. Gerstel, R. Casellas, A. Farrel, D. King, S. Lopez-Buedo, A. Cimmino, R. Morro, J. Fernandez-Palacios, Adaptive network manager: Coordinating operations in flex-grid networks, in: ICTON, June 2013.
[75] Telefonica, Telefonica & Infinera Network as a Service (naas) Demonstration Using Software Defined Networks,⟨https://www.infinera.com/wp-content/uploads/2015/08/Infinera-Telefonica_SDN_Demo.pdf⟩.
[76] ETSIs, NFV Proofs of Concept,⟨http://www.etsi.org/technologies-clusters/technologies/nfv/nfv-poc⟩.
[77] ETSI, Network Functions Virtualisation (NFV); Management and Orchestration, ⟨http://www.etsi.org/deliver/etsi_gs/NFV-MAN/001_099/001/01.01.01_60/gs_nfv-man001v010101p.pdf⟩, 2014.
[78] IETF, Network Function Virtualization Research Group,⟨https://irtf.org/nfvrg⟩, 2016.
[79] J. Soares, C. Goncalves, B. Parreira, P. Tavares, J. Carapinha, J.P. Barraca, R.L. Aguiar, S. Sargento, Toward a telco cloud environment for service functions, IEEE Commun. Mag. 53 (2) (2015).
[80] F.Z. Yousaf, T. Taleb, Fine-grained resource-aware virtual network function management for 5 g carrier cloud, IEEE Netw. 30 (2) (2016).
[81] Y. Chen, Y. Qin, M. Lambe, W. Chu, Realizing network function virtualization management and orchestration with model based open architecture, in: CNSM. IEEE, 2015.
[82] MEF, The third network: Lifecycle service orchestration vision,⟨https://www.mef.net/Assets/White_Papers/MEF_Third_Network_LSO_Vision_FINAL.pdf⟩, Nov. 2015.
[83] P. Quinn, U. Elzur, A Border Gateway Protocol 4 (BGP-4), Internet Draft, RFC, Jan. 2016. [Online]. Available:⟨https://tools.ietf.org/pdf/draft-ietf-sfc-nsh-02.pdf⟩.
[84] H. Li, Q. Wu, O. Huang, M. Boucadair, C. Jacquenet, W. Haeffner, S. Lee, R. Parker, L. Dunbar, A. Malis, J. Halpern, T. Reddy, P. Patil, Service Function Chaining (SFC) Control Plane Components and Requirements, Internet-Draft, Informational, Jan. 2016.
[85] R. Penno, P. Quinn, D. Zhou, J. Li, Yang Data Model for Service Function Chaining, Internet-Draft, Informational, Jan. 2016.
[86] OpenStack, Service Function Chaining Extension for Openstack Networking, ⟨http://docs.openstack.org/developer/networking-sfc/⟩.
[87] Open Network Foundation, ONF TS-027: L4-L7 Service Function Chaining Solution Architecture,⟨https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/L4-L7_Service_Function_Chaining_Solution_Architecture.pdf⟩, 2015.
[88] C. Filsfils, N.K. Nainar, C. Pignataro, J.C. Cardona, P. Francois, The Segment Routing Architecture, in: IEEE GLOBECOM, Dec 2015.
[89] C. Filsfils, S. Previdi, B. Decraene, S. Litkowski, R. Shakir, Segment Routing Architecture, Internet-Draft, Tech. Rep., Jun. 2016.
[90] S. Previdi, C. Filsfils, B. Field, I. Leung, J. Linkova, E. Aries, T. Kosugi, E. Vyncke, D. Lebrun, IPv6 Segment Routing Header (SRH), Internet-Draft, Tech. Rep., Mar. 2016.
[91] C. Filsfils, S. Previdi, A. Bashandy, B. Decraene, S. Litkowski, M. Horneffer, R. Shakir, J. Tantsura, E. Crabbe, Segment Routing with MPLS data plane, Internet-Draft, Tech. Rep., Mar. 2016.
[92] Open Network Operating System, CORD: reinventing central offices for efficiency and agility, ⟨http://opencord.org/⟩.

[93] CISCO, Cisco IOS XR Routing Configuration Guide - Segment Routing,⟨http://www.cisco.com/c/en/us/td/docs/routers/crs/software/crs_r5-2/routing/configuration/guide/b_routing_cg52xcrs/b_routing_cg52xcrs_chapter_0110.html#concept_360441CD7F564CF99C0E1765E15EB47F⟩.
[94] L. Velasco, A. Castro, D. King, O. Gerstel, R. Casellas, V. Lopez, In-operation network planning, IEEE Commun. Mag. 52 (1) (2014).
[95] E. A. Brewer, Towards robust distributed systems (abstract), in: PODC. ACM, 2000.
[96] S. Scott-Hayward, G. O'Callaghan, S. Sezer, SDN Security: A Survey, in: IEEE SDN4FNS, Nov 2013.
[97] ETSI, Network Functions Virtualisation (NFV); NFV Security; Privacy and Regulation; Report on Lawful Interception Implications,⟨http://www.etsi.org/deliver/etsi_gs/NFV-SEC/001_099/002/01.01.01_60/gs_NFV-SEC002v010101p.pdf⟩.
[98] OASIS TOSCA, TOSCA Simple Profile for Network Functions Virtualization (NFV) Version 1.0,⟨http://docs.oasis-open.org/tosca/tosca-nfv/v1.0/csd02/tosca-nfv-v1.0-csd02.pdf⟩, 2015.
[99] Bjorklund, YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF), RFC 6020, Oct. 2015. [Online]. Available:⟨https://rfc-editor.org/rfc/rfc6020.txt⟩.
[100] B. Claise, C. Moberg, YANG Model Classification, Internet Draft, RFC, Apr. 2016.
[101] Ansible, ⟨https://www.ansible.com⟩.
[102] Chef, ⟨https://www.chef.io⟩.