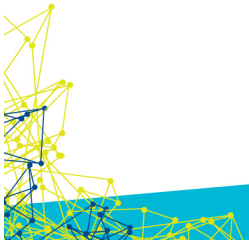
OPEN NETWORKING
FOUNDATION

TR-518
Relationship of SDN and NFV

Issue 1
October 2015



ONF Document Type: Technical Recommendation
ONF Document Name: Relationship of SDN and NFV

Disclaimer

Open Networking Foundation
2275 E. Bayshore Road, Suite 103, Palo Alto, CA 94303
www.opennetworking.org

Table of Contents

1 Introduction

1.1 Summary

Disclaimer – Neither the SDN nor the NFV community speaks with a single, completely consistent voice. It will be possible to adduce evidence against any assertion in this paper. The paper nevertheless attempts to present a recognizably mainstream view of each discipline.

This paper compares and contrasts SDN, whose architecture is defined in [SDN], and NFV, as described in the set of phase 1 deliverables of the ETSI NFV ISG, as listed in References clause 4.2. Only publicly available documents are used. It is recognized that NFV work will have moved on from phase 1 completion, but stable, public documents are deemed preferable to an attempt to track a moving target, some of whose documents may not be publicly available.

The ONF view of SDN is intentionally scoped very widely (see note), defining principles and a framework that encompass any number of specialized sub-scopes. One of the focused sub-scope use cases of SDN is Transport SDN, another is NFV. By defining NFV over a scope narrower than that of SDN, the ETSI NFV ISG has been able to specialize the functions and their interfaces beyond the level of detail that would be appropriate in a general SDN architecture. It is useful to consider the ways in which NFV interprets SDN principles, and which new principles are appropriate as the result of NFV's more focused scope.

> Note – An expansive perspective is encouraged by typical SDN architecture drawings that show recursion, generic resources, and business relationships. The narrower, but deeper, focus of NFV is equally apparent in its bounded archetypal figures.

To clarify the distinctions between SDN and NFV, the present paper often describes the disciplines as if they were separate. Indeed, an implementation may choose to create disjoint NFV and SDN domains. A great deal of information is important to both, however, and both must be coordinated to achieve overall business objectives. In addition, each offers functionality that can help the other avoid reinvention of the wheel. The present paper argues that disjoint partitioning fails to exploit the relative strengths of the two disciplines.

The existence and disadvantage of silos have been recognized problems for many years. A major opportunity of today's re-thinking of the overall communications space is the ability to avoid silos in the future network. Yet, almost everything, both in standards organizations and in open-source initiatives, is focused on point solutions that address existing or new silos. Arguably, the most important silo now being newly created arises from the view that SDN and NFV are somehow different, rather than overlapping aspects of a common endeavor.

Admittedly, SDN and NFV are not identical. Recognizing broad overlap and fuzzy boundaries, this paper asserts that the perceived differences are largely due to perspective, application, and to a great extent, terminology.

1.2 SDN perspective

The target reader of this analysis is not necessarily expected to be thoroughly familiar with [SDN] (see note). This clause provides a high-level introduction to the SDN architecture, as it pertains to NFV.

Note – A view from the NFV perspective is being developed in ETSI NFV ISG.

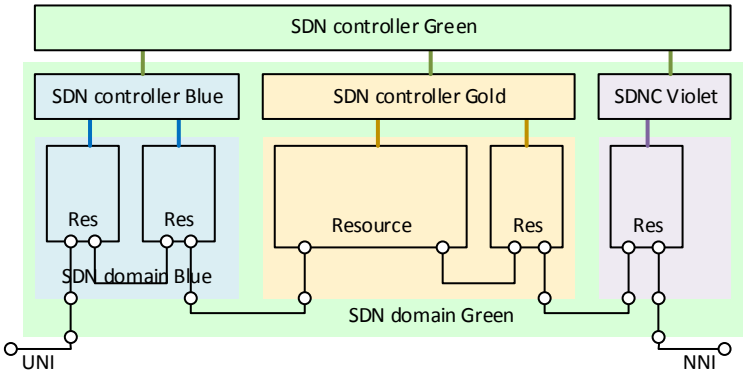


Figure 1 – SDN example

Figure 1 illustrates a global SDN domain, Green (see note), which offers service, for example from some given UNI to some particular NNI. The user of the service is understood to be a client (not shown) of SDN controller Green.

Note – In SDN documentation, color is generally used to distinguish domains, be they administrative, customer-provider, technology, or otherwise bounded. Especially for administrative and business domains, the interface is generally characterized by an SLA, information hiding, policy enforcement, and name/address space isolation.

The Green SDN controller accepts a service request from one of its clients and orchestrates it across a set of virtual resources offered by subordinate SDN domains Blue, Gold, and Violet. Handoff between domains (alignment of physical port, protocol stack, security, etc.) is the responsibility of the Green SDN controller, acting through Blue, Gold, and Violet SDN controllers. External handoff, in this example to the UNI and NNI, requires alignment between Green and the neighboring domains. Alignment may be achieved by provisioning, discovery, or negotiation.

A primary function of the SDN controller is to virtualize (see note) its underlying resources for the benefit of its clients, then to orchestrate the shared use of these resources on behalf of client demands. Both virtualization and orchestration are recursive and involve far more than simply subdividing or combining resources. See clause 3 and [SDN] for further discussion of SDN virtualization and orchestration.

Note – US English spelling is used throughout.

As mentioned, the Green SDN controller sees a set of resources for its exclusive use in satisfying the service request. Recursively within each subordinate domain, the local SDN controller likewise sees resources for its own use. The Blue SDN controller, for example, satisfies the Green service request by orchestrating its own available resources in light of a Green SLA (explicit or implicit), balanced with demand from other services. Having selected resource instances, the

Blue SDN controller sets up appropriate connectivity between them and provisions each instance with whatever specialized functions or parameters may be appropriate for that service. Using information provided by the Green SDN controller, the Blue SDN controller also provisions endpoint attributes for connectivity to the UNI and for handoff to the Gold domain.

An example of a specialized function might be a connectivity fault management (CFM) maintenance group end point (MEP), with parameters set for the particular service. Such a function might already be latent in the resource, but might also need to be downloaded as a code module before being initialized, provisioned and activated. Observe that the SDN controller thereby instantiates a new network resource, namely the MEP.

An SDN controller deals with generic resources. Some of the resources available to the Green SDN controller may be NFV virtual network functions (VNFs) or NFV network services (NSs). Figure 2 illustrates the possibility that some or all of the resources available to an SDN controller may be derived from NFV.

Following from the MEP example, and recognizing the generic and recursive nature of SDN resources, the SDN controller may instantiate a VNF of its choice on some available lower-layer container that it knows about.

This function clearly overlaps the capabilities of NFV. This scenario would logically be supported by making NFV functions available for invocation by the SDN controller, including the continuing life cycle management of the new VNF. Obvious variations include the use of pre-existing VNFs, requesting scaling in/out/up/down, etc.

A number of services may use a given resource simultaneously. For most services, a given resource requires per-service provisioning, for example customized filters in a firewall or parental control network function. When the resource is a VNF, this requires that the SDN controller know which VNF instance is in use for the given service and that it have management-control access to that VNF instance. This requirement limits the extent to which a group of independent resources, such as a VNF NS, can be exposed to the SDN controller as an opaque black box. Accordingly, figure 2 shows Gold and Violet SDN controllers with knowledge of particular VNF instances that may exist within their respective NFV domains.

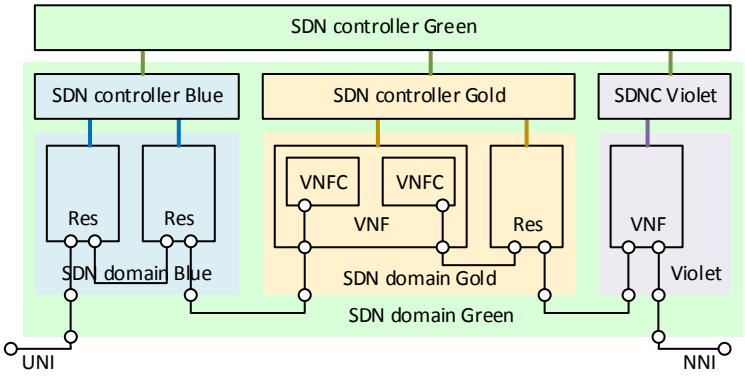


Figure 2 – Gold and Violet as NFV domains

In figure 2, the Gold and Violet SDN controllers are similar to the Blue SDN controller in orchestrating, interconnecting and provisioning resource instances. The Gold domain, however,

illustrates a VNF instance comprised of two VNFC instances. If only the composite VNF is exposed to the Gold SDN controller, then some other entity is responsible for interconnecting the VNFCIs. That other entity may also be an SDN controller (not shown). See figure 4 below for further discussion.

2 Discussion

2.1 Motivation

Our common purpose is to convey information amongst arbitrary entities, be the entities co-resident on the same physical platform or sited at opposite ends of the earth. Traffic may be discarded, stored and replayed, supplemented or modified, either in support of the communications process itself or in the form of added-value services. SDN focuses especially on the use of resources to provide services; NFV focuses especially on the creation and life cycle support of some classes of service resource.

The SDN business case is service fulfillment, faster and more flexibly executed than is possible under the previous mode of operation. To SDN, the idea of resource is generic; in principle, anything that can contribute to any kind of service is in scope. Some subset of the resources used by SDN may be supplied by NFV, whose business case is based on reducing the time and cost to provide a certain class of resources. NFV concepts fit best into the space of largely location independent software functions that can be executed on general purpose servers. NFV documents emphasize the creation and life cycle management of software resources, and their configuration and exposure for use beyond the NFV domain as VNFs or as VNF NSs. Subject to backhaul cost, the economics tends to favor hosting on COTS servers in data centers.

To an SDN controller, a VNF is just another resource, a node function in a network graph with known connectivity points and known and controllable transfer function. SDN service agility is enhanced by NFV's ability to rapidly create, scale or relocate virtual resources. Coordination is needed: SDN cannot use resources it does not know about, and NFV should not destroy resources that are in use. A common resource inventory will be important, including resource capability (e.g., firewall VNF), how to connect data and management-control channels, and current state.

NFV focuses on using comparatively inexpensive general-purpose computing and storage assets where feasible, to adapt the quantity and location of virtual resources to the need, and to avoid or minimize stranded compute, storage, and network capacity. NFV is motivated to replicate as much functionality as possible onto VNFs, migrate quickly, and retire legacy technology. However, not everything can be virtualized economically and immediately; legacy coexistence will be required for some transition period. SDN encompasses improvements to management and control, with gradual and incremental migration to new physical plant.

Both in terms of technology and because of physical connectivity, network resource instances are far less fungible than the servers that underlie NFV concepts. The cost of such specialized instances is a mandate that they be used efficiently. A high priority of SDN virtualization is therefore to define how resources can be shared statically or dynamically at any desired level of granularity, often varying over the course even of short intervals. In contrast, it is comparatively easy

and inexpensive to spin up a new VM, or even to add new servers to a PoP, so the cost-benefit tradeoff does not justify the sharing of sub-VNF functions in complex ways.

2.2 Core concepts

Concerned with delivery of services across a global span, and from end user interface all the way down to hardware settings, hierarchical and federated recursion are vital concepts to SDN as ways to deal with scale, business and trust boundaries, technology differences, legacy interworking and other factors. To date, the problems addressed by NFV have not required prioritization of recursion and federation as fundamental architectural concepts. Business boundaries are fundamental to SDN, but have not required as much focus in the NFV community, aspects such as contracted performance with verification, information hiding, policy enforcement, and name/address space isolation.

VNFs are often described as residing within a limited abstraction/recursion distance from physical hardware. To SDN, a VNF is just another resource, capable of residing at any suitable point in an infrastructure.

The network component of the NFVI is not an ideal resource with infinite bandwidth, zero contention, zero delay, zero failure rate, zero cost. As NFV enters real-world deployment, SDN concepts are positioned to evaluate and optimize operation in consideration of such factors.

Some network functions are tightly bound to the physical infrastructure. Such functions include, for example fate-shared OAM, protection switching engines, infrastructure load balancing, wavelength switching, and even the configuration of connectivity for NFV Management and Orchestration entities. These functions cannot be abstracted into VNFs, and necessarily lie in a part of the SDN domain that does not contain VNFs. [SDN] also explains how to map the ultimate implementation of such functions in hardware through multiple levels of virtualization, invisibly to the client. These tools will be useful to isolate and instrument services that incorporate VNFs.

2.3 Operation

When presented with a service request, an SDN controller orchestrates network services across non-NFV resources and VNFs, and optionally (see note) VNF network services (NSs). It builds the selected resources into the end-to-end service it constructs on behalf of its client or customer.

> Note – To provision service-specific parameters, an SDN controller requires access to specific VNF instances, which may not be exposed as VNF NSs. VNF NSs may nevertheless be useful as SDN resources if service-specific parameters are not needed or can be provided in some other way, for example by way of metadata added at an exposed SFC classifier, whose interpretation is statically known by the VNFs.

While an SDN controller must perform per-service-instance configuration of specific operational parameters, i.e., customer- or flow-specific attributes for everything from tag values to customized filter criteria, the NFV focus on life cycle maintenance implies resource initialization with globally appropriate attribute values, but generally not customization for each service instance in which the resource may be used.

An SDN controller normally needs to interwork with other domains, ensuring proper service monitoring and handoff, for example to CPE, to a UE, to a foreign administration, or to a disjoint

NFV or SDN domain. This means that handoff attributes must be known, for example through business agreement. Automating these handoff attributes (see note) is a major function of an SDN controller. To the extent that they share handoff points, a hypothetically separate NFV domain manager would cooperate with an SDN controller on this function.

> Note – Examples of handoff attributes include agreement on physical port, wavelength or time slot assignment, protocol stack with parameter values such as S-VID, MEP level and timing, encryption policy and parameters, authentication policy such as 802.1X, key management policy.

If needed for the service it is constructing, an SDN controller may download, install, and configure applets (i.e., VNFs (see note)) on particular platforms at appropriate topological points. NFV concepts and tools may prove valuable for at least some aspects of this.

> Note – The development of protocol-independent forwarding (PIF) models in ONF is recognized as an emerging way to define (V)NFs of particular types. An SDN controller will be responsible for instantiating such (V)NFs on suitable platforms, be they customizable hardware, fully software programmable, or anything in between. See further discussion in clause 2.8.

2.4 Dynamic behavior

Some end-user service demands may be highly dynamic, for example the arbitrary attachment of a roaming device, which may require authentication, retrieval of a subscriber profile, orchestration and provisioning of resource instances, possibly through a service chain, all essentially in real time and perhaps persisting for only seconds or minutes.

As well as rapid response, continuity in time is an important aspect of SDN, as customers churn, traffic flows come and go, and resources fail and are protected, repaired and built out. Figure 3 sketches the essence of an SDN controller, which is to act as the intelligent node in a feedback loop that continuously converges actual resource state toward desired state, constrained and prioritized by policy.

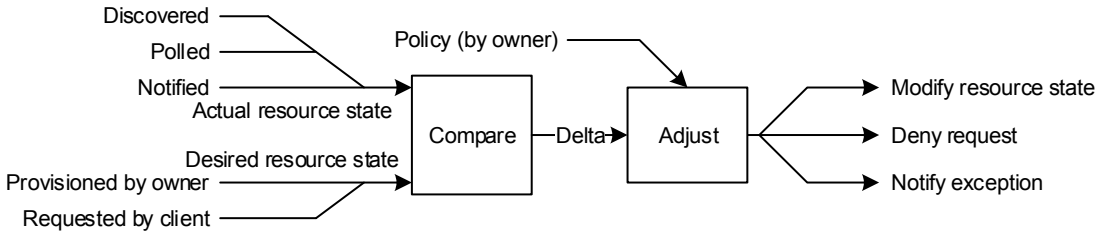


Figure 3 – Core function of an SDN controller

The idea of state is to be understood very broadly, including everything from the very existence of resources, to configured or observed parameter values, to indirect consequent behavior. It is not necessarily the case that actual and desired state map 1:1, or even directly, to each other; the Compare function includes whatever is needed to compare actual and desired state universes, and derive optimal, or near-optimal, adjustments according to policy. (The policy may also change from time to time.)

Some of the actual and desired state inputs may be generated by NFV entities, and some of the consequent actions may be requests to NFV entities for action. Notifications may be published by both SDN and NFV entities.

Any number of additional and auxiliary functions may be incorporated into an SDN controller as needed to achieve its core purpose or in support of additional features.

Nested and collaborative feedback loops are well understood in principle, as are the risks of instability, e.g., caused by underdamped loop gain or unintended dependencies between separate feedback loops.

Some of the options available to an SDN controller include re-partitioning or re-allocating existing resources, re-balancing traffic on existing resources, creating new resources (see note), or requesting the migration of existing resources to better fit demand to capacity. Clearly, services available from NFV components will be valuable to an SDN controller with these needs.

> Note – The statement that the SDN controller does X includes the highly desirable case that the SDN controller seamlessly invokes a service offered by an NFV component. Likewise, although NFV could invoke SDN functions across a formal silo boundary, it would be preferable to simply call a common library function.

2.5 Joint deployment

If multiple distinct SDN and/or NFV domains exist in a given network, it will be important to ensure that their activities do not work at cross purposes. An explicit SDN-NFV domain manager boundary might assist in avoiding dependency loops, but will carry an efficiency penalty. In any event, a great deal of shared information will be needed across domains, and mutual comprehension, write privilege, and synchronization will be important issues. It will be necessary that all disjoint entities publish and subscribe to notifications of common interest, so a common notifications framework will be important.

For VNF life cycle management, NFV documents identify the need for a priori information about instantiation, presumably including information such as initialization and port details (see note). As a resource user, SDN needs to know the functionality of available or potential VNFs, how to connect them (or their composites) into data plane services, and how to access them for control. Updates to VNF connectivity or internal state must be coordinated.

> Note – If a VNF has multiple data plane ports, they may be expected to fall into some number of equivalence classes. The meaning of each equivalence class must be known to any entity that uses the VNF, along with how to connect to ports in that equivalence class.

Some parts of an NFVI may be physically dedicated for use by a given NFV domain, but many resources, especially in the WAN, will be shared with other NFV or non-NFV domains, and in particular with SDN. The NFV and SDN domains must coordinate their claims to the shared resources, even dynamically on a packet-by-packet basis. Figure 4 illustrates how this might work. It is not necessarily intended that the NFV management entity (here generically designated an NFV manager) be separate from the SDN controllers, only to show the necessary functions. Ideally, an entity in either domain would simply make library calls to the functional specialties of the other.

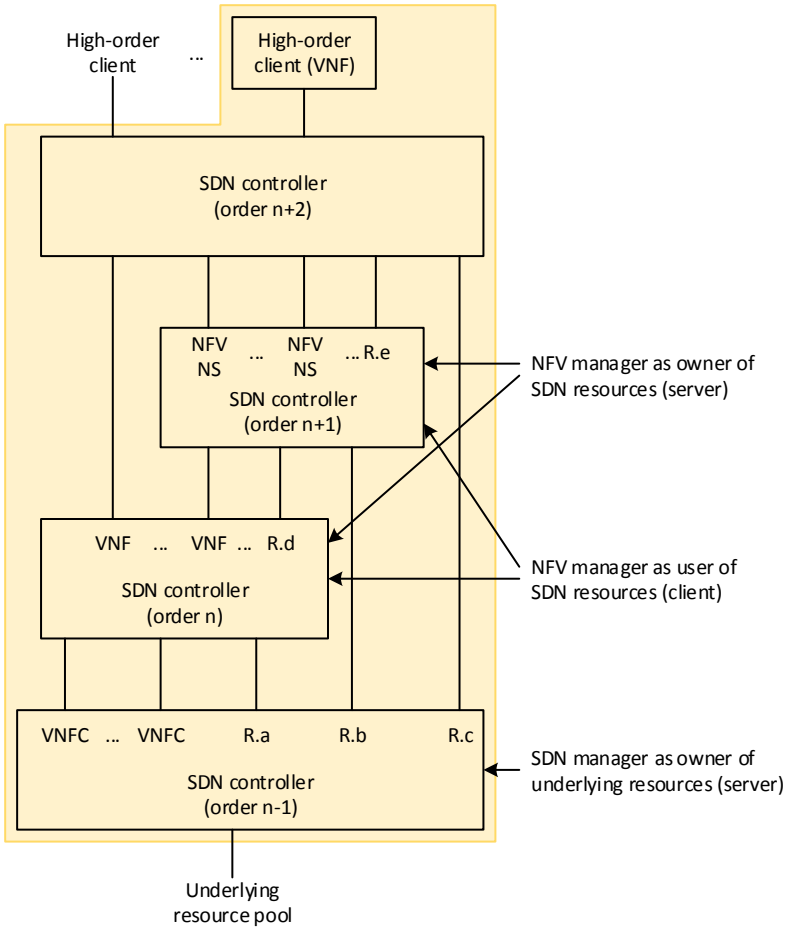


Figure 4 – Complex client-server relationships

Figure 4 illustrates how SDN controllers may be both servers and clients to an NFV domain shown in gold. The notation R.x is intended to indicate non-VNF/VNFC resources that may be needed to satisfy service demands.

At the top of the figure, high-order clients request network services from SDN controller (n+2). One of them is itself a gold VNF.

SDN controller (n+2), which is itself imagined as a gold VNF (see note), satisfies the service request by provisioning service-specific attributes into its available resources, some of which may be NFV network services, some of which may be VNFs, and some of which may be other resources, including network resources that are used to interconnect the components of the particular service.

Note – It is vital to avoid dependency loops in assigning responsibilities.

Suppose that SDN controller (n+2) requires a resource that does not exist or is otherwise inadequate for some reason. If allowed by policy, SDN controller (n+2) may request a new or scaled NS, which may in turn trigger SDN controller (n+1) to request a new or scaled VNF resource from SDN controller (n). SDN controller (n+2) may also request a new or scaled VNF resource

directly from SDN controller (n). When the resource is made available, SDN controller (n+2) completes the service request by provisioning service-specific attributes in the new resource.

At whatever level of abstraction, an SDN controller can and should invoke operations from the NFV discipline to create or scale the necessary resources. Interconnection of the resources implies that the NFV manager can and should reciprocally invoke SDN control at a less abstract level. Interworking at the level of library calls is encouraged because it would avoid domain separation and a protocol interface.

This example illustrates a top-down service-driven resource request push through the SDN domain. It is equally possible that some other impetus, for example network planning, could drive the process of creating resources from the bottom up. In this case, the NFV manager could first instantiate VNFCs or VNFs, as the case may be, invoke an SDN controller to interconnect them, and then expose the resulting new resource to an SDN controller at a higher level of abstraction.

2.6 Information modeling

Both SDN and NFV communities anticipate that the communications environment of the future will include more vendors with a wide range of product offerings, on short and unsynchronized release schedules. Especially if a service invocation goes through several APIs from several vendors before reaching a data-plane device, semantic mismatches in the information conveyed will clearly result in chaos. In addition, both SDN and NFV communities accept the need to coexist with the present OSS environment, at least in the near term.

The importance of a common information model [CIM] cannot be overstated (see note). The risk of mismatch is exacerbated if SDN and NFV standards, products and environments are separate. To a certain extent, the difference in focus reduces the risk: SDN is concerned with the details of resources as they are used for services, NFV with software installation and maintenance; but it behooves the communities to work together to avoid discontinuities at the overlap points.

> Note – Purpose-specific data models can be derived from the common information model as necessary. This level of detail is comparatively easy to adapt pragmatically, as long as semantics are preserved. This allows purpose-specific APIs to be produced, while enforcing common semantics for the resources of concern.

2.7 The role of OSS

In [SDN], the presence of an OSS or management block is recognition that some of the necessary functionality of network operations is beyond the current ambition level. Examples of such functions include equipment and software installation and upgrade, fault management and troubleshooting. To SDN, such additional functions are a matter of time and priority, not a declared scope boundary. In particular, many of today's OSS functions are within the long-term scope of SDN, for example service order negotiation and fulfillment, and inventory maintenance.

NFV drawings show both an OSS/BSS block and an EMS. Many FCAPS functions have not been addressed, and per-service provisioning is often considered to be an EMS function. It is unclear whether the long-term view of the NFV community envisions superseding the EMS function and blending seamlessly with the OSS/BSS.

Coordination between SDN and NFV disciplines will be necessary as each does or does not extend its scope. SDN architectural evolution is aligned with many of the future mode of operations (FMO) ideas in the TMF ZOOM project, in particular the idea of management-control continuum (MCC).

2.8 Further work

This paper identifies a number of areas where SDN and NFV perform the same functions or are in a position to offer services to each other. They must inevitably share a great deal of common information, either static or dynamic. As well as information model alignment, it will be important to agree on a number of related topics such as security, shared write privileges if needed, publish and subscribe mechanisms. This large project should be done collaboratively.

It may be possible to extend some areas of overlap in ways that add value to the wider space. From the SDN perspective, the most obvious example is life cycle management. NFV focuses on installing executable code in software containers, and it makes sense for SDN to use these tools in that situation. However, the same concepts and tools would be very useful as a way to instantiate more general resources, in which the download and initialization could be onto a specialized hardware platform, rather than a software container, and could be an opaque block of configuration data, rather than any sort of executable image.

Arguing that they present few or no concerns unique to SDN, [SDN] intentionally omits distributed information synchronization, reliability and availability from the SDN architecture. If NFV concepts and tools could be applied directly to SDN controllers or other parts of the SDN domain, they would be highly valued. In NFV-rich environments, it may be expected that most, if not all, SDN controllers would be implemented as VNFs.

If NFV and SDN were to be implemented as separate but collaborating domains, it would be necessary to formalize interfaces through which either could query or invoke the services of the other. Perhaps the ideal outcome for shared responsibility would be libraries of mutually useful functions that could be built into any product configuration desired, be it an SDN controller or any of the NFV entities.

Where code itself could not be re-used, it might still be possible to use common APIs, with the underlying code unique to the pertinent use cases. Failing that, each discipline might be able to exploit the principles of the other, in a way to achieve alignment at higher levels of abstraction, for example parallel governance of bodies of code, resource inventory, or other assets.

Additional avenues to achieve synergy should be actively pursued.

For any of these to happen, the disciplines need to understand each other’s concepts and terminology in depth, and actively seek to combine comparative strengths to resolve comparative weaknesses. This will be a continuing effort.

3 Appendix – Terminology

3.1 Introduction

Understanding is a necessary precondition of agreement.

More heat than light arises from discussions in which the parties are talking about different things, especially when they do not realize it. This situation can arise in several ways.

- The same word may have different meanings to different participants. Often, a term is poorly defined or left undefined under the expectation that the entire community understands its meaning through common usage. This may suffice within a community with a rich shared history and outlook, but it complicates communication with other communities, and raises the cost for newcomers to join the original community.
- Some or all of the same semantics may be encompassed under different terminology. The best example in the current context is perhaps the overlap between management and control, discussed below.
- It can be hard to distinguish different views of the same underlying concepts when they appear in several valid but different contexts as illustrated in e.g. Figure 4, where resources and controllers appear at several levels of abstraction and VNFs are both the homes of functions and the functions themselves. (And Figure 4 does not even mention the case when MANO components are themselves VNFs.)
- Perspective matters. From some viewpoints, a VPN, E-LINE, PVC, cross-connection and a tunnel are substantially identical; from other viewpoints, they differ significantly.
- Focus matters. [SDN] intentionally takes an expansive view of the space, defining principles and major blocks while intentionally underspecifying detail. This allows any number of arrangements to claim (correctly) to comply with the principles of the SDN architecture. NFV takes an expansive view of a different problem space with a tight focus on a subset of the SDN problem space. As such, the idea of compliance differs between communities.

Both speaker and listener (respectively writer and reader) have a responsibility to be alert to mismatches of understanding. In case of doubt, the speaker must make his definition clear. Failing that, the listener may make good-faith deductions, but should explain his reasoning for validation by the speaker. Virtualization is perhaps the best example of such a term.

3.2 Virtualization

Virtualization has quite distinct meanings in the SDN and NFV communities. Consider first the SDN definitions:

> [SDN], emphasis added: *An abstraction is a representation of* [one or more (see note)] *entity*[s] *in terms of selected characteristics, while hiding or summarizing characteristics irrelevant to the selection criteria.*
>
> > Note – The concept allows combining characteristics from several underlying entities into the new abstract entity, not merely decomposing a single entity. This clarification needs to become part of [SDN] issue 2.
>
> *… a virtualization is an abstraction whose selection criterion is dedication of resources to a particular client or application.…*

Several consequences follow from this conceptualization.

1. All views are abstract, a purposeful view is virtual. Proximity to a physical substrate is simply a special case.
2. There is no atomic unit of granularity. Virtualization models entities at the level of detail appropriate to its purpose.
3. Recursion is natural: an (SDN) virtualization can be further (SDN) virtualized in as many ways, and to whatever depth as may be desired.
4. Resources of any type may be partitioned and combined arbitrarily in an (SDN) virtualization. The properties of the (SDN) virtual entity may differ completely from the properties of any of its components.
5. The resources contributing to an (SDN) virtualization may individually exist at differing levels of (SDN) virtualization.
6. Separate (SDN) virtualizations over the same resource pool need not be contained, disjoint, or have any other particular relationship, except to the extent that they are designed not to contend for common underlying resources.

In NFV documentation ([NFVI-OV] clause 6.1.1), (NFV) virtual always refers to a software entity in a container, which is typically understood to be a VM over a hypervisor on a COTS server. OS containers and JVMs also exemplify the idea.

> [VR], emphasis added: *4.1 Introduction*
>
> *Virtualisation aims to transform the way that network operators architect networks by evolving existing IT virtualisation technology and making use of cloud computing techniques in order to consolidate network equipment onto industry standard high volume servers, switches and storage, which could be located in N-PoPs, Network Nodes and in the end user premises.*
>
> *Virtualisation involves the implementation of network functions in software that can run on a range of industry standard hardware, enabling ubiquitous, convenient and on-demand access to a shared pool of configurable computing resources (e.g., networks, servers, storage and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.*
>
> [NFVI-OV], *3.1 Definitions*

> ***Virtual network****: topological component used to affect [sic] forwarding of specific characteristic information*
>
> *NOTE 1: The virtual network is bounded by its set of permissible network interfaces.*
>
> *NOTE 2: In the NFVI architecture, a virtual network forwards information among the network interfaces of VM instances and physical network interfaces, providing the necessary connectivity and ensures secure isolation of traffic from different virtual networks.*

The definition encompasses any network, virtual or not, as long as it is restricted to traffic forwarding. The NFVI network is focused on providing connectivity services for VNFs. In SDN, a network includes nodes that may process traffic as well as those that only forward traffic.

SDN virtualization is primarily directed at representation of resources (e.g., a transport network) in customized ways, whereas NFV virtualization is directed at separation of functionality from infrastructure. It will be apparent that (SDN) virtualization is a superset of (NFV) virtualization, and that the idea of a virtual network function is compatible with both environments.

3.3 Orchestration

Although [SDN] does not formally define (SDN) orchestration, the meaning of the concept is apparent from the following excerpts:

> *4.3.1: An SDN controller is expected to coordinate a number of interrelated resources, often distributed across a number of subordinate platforms, and sometimes to assure transactional integrity as part of the process. This is commonly called orchestration. An orchestrator is sometimes considered to be an SDN controller in its own right, but the reduced scope of a lower level controller does not eliminate the need for the lower level SDN controller to perform orchestration across its own domain of control.*
>
> *4.3.3: Because the scope of an SDN controller is expected to span multiple (virtual) NEs or even multiple virtual networks (with a distinct D-CPI* [data-control plane interface] *instance to each), the DPCF* [data plane control function] *must include a function that operates on the aggregate. This function is commonly called orchestration. This architecture does not specify orchestration as a distinct functional component.*
>
> *4.4: An SDN application may invoke other external services, and may orchestrate any number of* [additional] *SDN controllers to achieve its objectives. The OSS link and the coordinator function* [in the associated figure] *recognize that, like the other major blocks of the architecture, SDN applications require at least a certain amount of a priori knowledge of their environments and roles.*

A provisional SDN definition of (SDN) orchestration might be: the continuing process of allocating resources to satisfy contending demands in an optimal manner. The idea of optimal would include at least prioritized customer SLA commitments, and factors such as customer endpoint location, geographic or topological proximity, delay, aggregate or fine-grained load, monetary cost, fate-sharing or affinity. The word *continuing* incorporates recognition that the environment and the service demands constantly change over the course of time, so that orchestration is a continuous, multi-dimensional optimization feedback loop.

NFV does not define orchestration explicitly. Its meaning may be inferred from the NFVO definition [TERMS]:

> ***Network Functions Virtualisation Orchestrator (NFVO)****: functional block that manages the Network Service (NS) lifecycle and coordinates the management of NS lifecycle, VNF lifecycle (supported by the VNFM) and NFVI resources (supported by the VIM) to ensure an optimized allocation of the necessary resources and connectivity.*

Where lifecycle management is defined as:

> ***Lifecycle management****: set of functions required to manage the instantiation, maintenance and termination of a VNF or NS.*

Although NFV speaks of (NFV) orchestration elsewhere, for example among the responsibilities of the VIM ([MANO] 5.4.3 bullet 1), it usually thinks of (NFV) orchestration as a single concentrated functional block, without delegation. The NFV orchestrator may consider resource availability and load when it responds to a new demand, and may rebalance capacity as needed, including creating, deleting, scaling and migrating VNFs.

Except for explicit delegation, many of these are the same as the responsibilities of an SDN controller, although NFV emphasizes resource life cycle, while SDN focuses more on timely service fulfilment. The absence of well-defined principles for delegation limits the ability of an NFV environment to scale. Nor is it well defined how an NFV environment can share or delegate orchestration functions with an SDN environment. As scale and multi-domain use cases appear, it is safe to predict that NFV will invent delegation. SDN concepts may be helpful.

3.4 Control and management

NFV documents use the word *management* extensively while *control* is mainly found in NFVI [NFVI-ND]. In SDN, *control* is the operative term, and *management* is sometimes a sideline (see note). To a considerable extent, this reflects a traditional distinction between life cycle operations (NFV focus) and real-time service operations (SDN focus).

> Note – Even though many of the functions expected from an SDN controller are today performed by Element or Network *Management* Systems.

To introduce the topic in an SDN context, recognize that, to properly perform their functions, virtualizer and orchestrator must be configured. The entity that performs this function must have a full view of and full power over all data plane resources, and full view of commitments to customers (aka clients, applications, tenants (see note)). [SDN] calls this entity a manager or OSS, to be provided by the effective owner of the underlying resources and the controller.

> Note – The term *tenant* suggests occupancy, in some sense, of resources that are owned by a *landlord*. If a customer application were hosted on a provider server, the idea of tenancy would be applicable. However, the occupancy implication is usually irrelevant and in SDN provider-customer relations, rarely true, so other terms are preferred in SDN.

Once resources are dedicated to a client, the client becomes their effective owner, and among other things, can further allocate them to its own customers. The SDN community and [SDN] issue 1 refer to functions exercised by a customer, client, or app, as control.

The original idea was that a client or a customer would use its resources to realize network services directly. But a given SDN controller has no way to know whether its customer is using resources to realize services directly or is further allocating those and other resources to its own customers. The purpose for which the resources are used is local to the customer domain; underlying resources are simply commanded to perform in a given way, with explanations neither required nor offered.

Recognizing that hierarchy is only visible from the viewpoint of the gods, [SDN] issue 2 is planned to describe interfaces and functions from the local perspective, but in a way that supports recursion. The important distinction is between underlying resources, i.e., south of the controller, and (further) virtualized resources exposed to clients. According to this formulation, the manager-OSS and the client perform similar functions, but with different scopes.

This aligns with the emerging concept of MCC (management-control continuum), which likewise asserts that there are no hard criteria that distinguish management from control, merely differences in scope, perspective, and time frame.

For future use, the term *control* is recommended, because its existing connotation more naturally encompasses the ideas of continuing feedback response and optimization in real time, along with peer-peer interactions (e.g., signaling, route discovery) that are not discussed above. Both terms will likely remain common, used where their conventional connotation best fits the context. However, any implied differences between management and control should be interpreted cautiously.

3.5 Domains

A domain is simply a grouping of entities according to some criterion. Domains can grow and shrink over the lifetime of the group, as changing candidate entities do or do not satisfy the membership criterion.

> Note – The definition in [NFVI-OV] reads:
>
> *domain: specific part of a larger entity which is useful to separate out based on given criteria.*

The term *SDN domain* refers to the set of resources controlled by a given SDN controller. The term may also be used in context to include the SDN controller, recognizing both data plane and control-management functions.

In this document, the term *NFV domain* refers to a similar concept in the NFV world, expecting that implementations of MANO, VNF and NFVI entities would be grouped together. It would be possible for an NFV PoP to be a domain, for example. As with SDN domain, the term *NFV domain* is often used to include both data plane and control-management functions.

3.6 Point of presence (PoP)

The pertinent definitions in [NFVI-OV] read:

> *NFVI-PoP: single geographic location where a number of NFVI-Nodes are sited.*

NFVI-Node: physical device deployed and managed as a single entity providing the NFVI functions required to support the execution environment for VNFs.

Observations:

- This definition is understood to include customer premises, where the number of NFVI nodes could be as few as one.
- Many network resources cannot exist in an NFV PoP. These include
  - Physical network functions and local links, because they do not support a VNF execution environment (depending on how *support* and *execution environment* are to be understood)
  - Links and subnetworks that span more than one geographic location

The SDN architecture is abstract and functional, and does not use the idea of PoPs, although [SDN] recognizes more generalized location considerations as one criterion for optimally selecting resource instances to fulfill a given service request. Two examples illustrate this point. First, proper choice of geographic or topological location can minimize transport cost, complexity, or latency. Second, physical grouping may be important to meet service availability commitments, for example when configuring dual homing or avoiding common cables or rights of way.

4 Back material

4.1 Acronyms

BSS	Business support system
CFM	Connectivity fault management
COTS	Commercial off-the-shelf
CPE	Customer premises equipment
DPCF	Data plane control function
EMS	Element management system
ETSI	European Telecommunications Standards Institute
FCAPS	Fault, configuration, accounting, performance, security [management]
FMO	Future mode of operation
ISG	Industry Specification Group
JVM	Java virtual machine
MANO	Management and orchestration
MCC	Management-control continuum
MEP	Maintenance group endpoint
NFV	Network Functions Virtualization
NFVI	NFV infrastructure
NFVO	NFV orchestrator
NNI	Network-network interface
NS	Network service
OAM	Operations, administration, maintenance
OS	Operating system
OSS	Operations support system
PIF	Protocol-independent forwarding
PoP	Point of presence
PNF	Physical network function
PVC	Permanent virtual circuit
SDN	Software-defined networking
SFC	Service function chaining
SLA	Service level agreement
S-VID	Service VLAN identifier
UE	User equipment
UNI	User-network interface
VIM	Virtual infrastructure manager
VLAN	Virtual local area network
VM	Virtual machine
VNF	Virtual network function
VNFC	VNF component
VNFCI	VNFC instance
VNFM	VNF manager
VPN	Virtual private network
WAN	Wide-area network

ZOOM Zero-touch orchestration, operations & management

4.2 References

[USE] ETSI NFV GS NFV 001 V1.1.1, Use Cases, 2013

[N-ARCH] ETSI NFV GS NFV 002 V1.2.1, Architectural Framework, 2014

[TERMS] ETSI NFV GS NFV 003 V1.2.1, Terminology for Main Concepts in NFV, 2014

[VR] ETSI NFV GS NFV 004 V1.1.1, Virtualisation Requirements, 2013

[NFVI-OV] ETSI NFV GS NFV-INF 001 V1.1.1, Network Functions Virtualisation (NFV); Infrastructure Overview, 2015

[NFVI-ND] ETSI NFV GS NFV-INF 005 V1.1.1, NVF Infrastructure; Network domain, 2014

[NFVI-METH] ETSI NFV GS NFV-INF 007 V1.1.1, Network Functions Virtualisation (NFV); Infrastructure; Methodology to describe Interfaces and Abstractions, 2014

[MANO] ETSI NFV GS NFV-MAN 001 V1.1.1, Management and Orchestration, 2014

[SWA] ETSI NFV GS NFV-SWA 001 V1.1.1, Virtual Network Functions Architecture, 2014

[SDN] ONF TR-502, SDN architecture, Issue 1, 2014

[CIM] ONF TR-513, Common information model overview, Version 1.0, 2015

