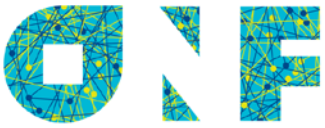

OPEN NETWORKING FOUNDATION

Framework for SDN: Scope and Requirements

Version 1.0
June 2015

ONF TR-516

ONF Document Type: Technical Recommendation
ONF Document Name: Framework for SDN: Scope and Requirements

Disclaimer

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

Table of Contents

1. Overview

1.1 Goals for Framework Document and Related Documents

According to the 2013 Open Networking Foundation (ONF) Architecture Working Group charter, the following points need to be addressed in the Software Defined Networking (SDN) Framework document (i.e. this document):

1. *Define the broad problem(s) that need to be solved and why these need to be solved, referencing and incorporating material on use cases from the market and education working group.*
2. *Define the requirements that need to be met by the architecture (specifically in areas of applicability, scalability, features and fault tolerance).*
3. *Define what is in scope and out of scope (e.g. [whether SDN] controller to controller communication [is] in scope or out of scope).*
4. *Define the terminology for SDN.*

This document accordingly lists illustrative usage scenarios, defines the problems to be addressed and the scope to be covered by the SDN Architecture document, and concludes by specifying the requirements for the SDN Architecture.

This revision of the Framework document should be read in conjunction with issue 1 of the Architecture document [2]. While Issue 1 of the Architecture document addresses many of the problems and requirements mentioned in the Framework document, some items remain to be addressed in future revisions of the Architecture document and/or in other ONF documents.

The Framework document only defines the terms used in the Framework document itself. A more extensive set of terms and their definitions can be found in the Architecture document.

1.2 Intended Audience and Lifetime

The SDN Framework and SDN Architecture documents are intended to describe ONF's view with respect to the extent of the SDN Architecture to all interested parties. They also supply guidance to individual ONF Projects, Discussion Groups and Working Groups with respect to the scope of SDN, the preferred architectural approach, and key requirements. Detailed specification of architectures, technologies and standards (e.g. individual behavior and interface specifications) is the purview of specific working groups. While the Framework and Architecture documents are intended to be of use for a number of years, periodic updates are expected to reflect the evolution of technology and use cases.

1.3 Summary

1.3.1 Connotations of the Term "Software Defined Networking"

Software Defined Networking (SDN) is a marketing term that generally refers to permitting software to interact with a network, with the details of the supported interactions varying, and with the interfaces that permit interactions not necessarily conforming to standards. Standardized SDN, where interfaces and behavior are standardized, forms a subset of SDN. SDN as discussed and standardized within the ONF is furthermore a subset of standardized SDN. Figure 1 below depicts these distinct connotations of the term SDN.

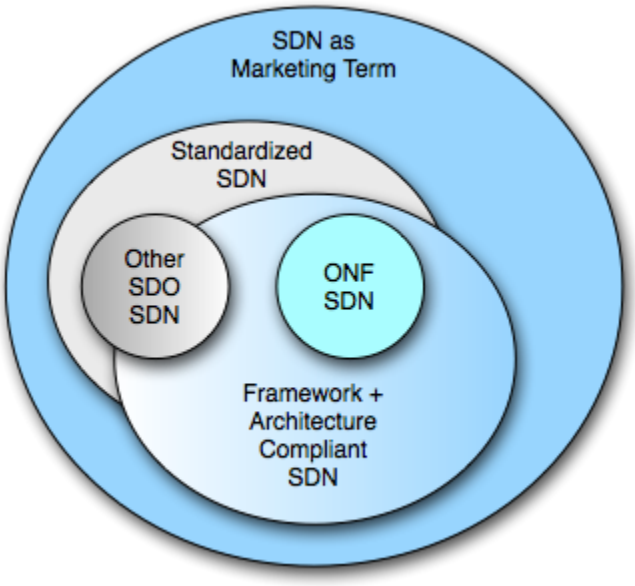

Figure 1: Framework + Architecture Compliant SDN vs. Other SDN

1.3.2 Key Principles of Software Defined Networking

The key principles of Software Defined Networking (SDN) are:

- SDN enables programmatic and abstracted interaction with the network. The classes of interaction that need to be considered include control, provisioning, configuration, management, and/or monitoring.
- The network control function is decoupled from the controlled Network Elements while being logically centralized. The control function and selected management/monitoring functions are housed in an SDN Controller.
- The SDN Controller is required to be able to exert programmatic direct control of forwarding behavior, thereby actually defining the network (e.g. its logical topology), not merely influencing/configuring it.

Any technology that conforms to the principles and meets the requirements elucidated in the SDN Framework and Architecture documents is accepted as meeting the definition of SDN embodied in these documents, whether or not the technology is standardized (by ONF or others).

Figure 2 below depicts these principles.

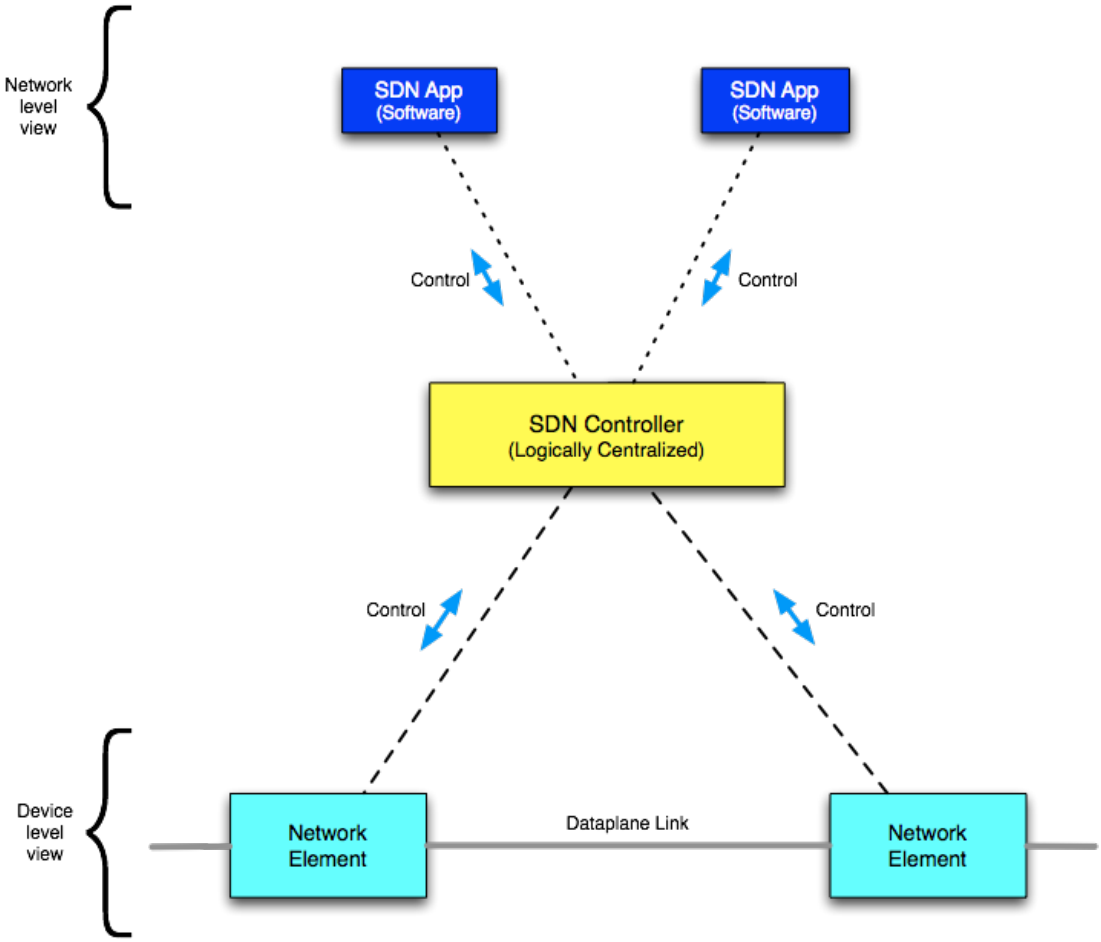

Figure 2: Programmatic Abstracted Interaction with Network

The currently defined OF-Switch and OF-Config specifications are examples of interfaces that, although they are still evolving, conform to the principles and requirements of the SDN Framework and Architecture. Other instances of SDN interfaces (for example Northbound Interfaces) that are conformant can also be contemplated and standardized by the ONF or other Standards Development Organizations (SDOs). Where possible, the number of standardized interfaces should be limited to a small number (ideally one) in order to facilitate interoperability and eliminate the need for interface adapters.

1.3.3 Scope of SDN Framework and Architecture

The scope of the SDN Framework and Architecture is summarized below.

1. Controller Plane - Data Plane Decoupling (Logical Separation):
    a. Distributed, hierarchical and federated deployment of SDN Controller instances, with Intra-Domain and Inter-Domain connectivity between SDN Controller instances;
    b. Northbound Interfaces (NBIs), enabling the development of SDN Clients (within applications) that interact with the SDN Controller plane;
    c. Permitting abstract views of the network to be created, interrogated and

manipulated. These abstract views can exist at various levels, and they can be combined with virtualization to create sliced or partitioned views. Furthermore, a comprehensive view may encompass all aspects of a network, whereas a more limited view may focus on specific aspects, e.g. security, reliability, quality of service, etc.

2. Network Elements:
    a. Modeling of forwarding and processing behavior, where *processing* includes computation, storage and/or physical or virtual network functions (NFs), whether these are directly related to traffic flows or whether they are associated with networking in general (e.g. processing of aggregated statistics or monitoring of events);
    b. Support for a variety of media and connectivity types (e.g. packet and circuit switching technologies);
    c. Support for diverse hardware and software platforms, with varying port counts, throughput, capacity, and flexibility, e.g. platforms supporting a fixed set of protocols (i.e. pre-defined protocol parsing and fixed matching/action pipelines), platforms supporting protocol independent forwarding (configurable parsing, matching and actions), as well as platforms supporting stateful classification and processing.
    d. Here a Network Element is defined as a collection of resources managed as a unit, i.e. a network node, at any level of abstraction, not necessarily a single "box" in the network.
3. Network Virtualization related technologies:
    a. Support for static and dynamic network provisioning technologies;
    b. Support for tunnels – used to create overlay networks.
4. Support for Network Functions Virtualization (NFV), L4-L7 Services, and data plane services:
    a. Identifying relevant traffic and diverting it to / through individual services, service chains or Virtual Network Function forwarding graphs;
    b. Service specific forwarding, e.g. application aware QoS, or service specific processing, such as encryption and content replication.
    c. Managing services and VNFs.
5. Enabling traditional and SDN controlled networks to coexist ("Hybrid" scenario):
    a. Partitioning elements with some ports being traditionally controlled, others SDN controlled;
    b. SDN control of an overlay network, with traditional control of the underlying network, and vice versa;
    c. Partitioning of responsibilities, for example, traditional management/monitoring and SDN forwarding configuration or vice versa;
    d. Enabling traditionally controlled and SDN controlled networks to be interconnected (for both intra-domain and inter-domain scenarios).

Requirements and criteria to evaluate the SDN Architecture are defined. These can be grouped in the following categories:

1. Simplicity and Expressiveness
2. Applicability
3. Interworking
4. Interoperability
5. Scalability
6. Security

7. Resilience and Fault Tolerance
8. Management and Monitoring

For more information on each category, refer to Section 4 below.

The Architecture specifies, at a high level, the reference points and interfaces used in a Software Defined Network, including those pertaining to the SDN Controller and the SDN controlled Network Element. Functional blocks internal to the Controller and Network Element may be mentioned to aid the description, however such functions are not necessarily required in each implementation, and the interfaces to such blocks are typically not specified.

When describing the behavior of the SDN Controller / Network Element, the focus is on aspects affecting interoperability. Decisions are made by the Architecture with respect to component partitioning, as well as which interfaces are described and which are standardized, with a rationale being supplied. The focus is on principles, with other working groups defining details, for example specific interfaces.

Common models and mechanisms are recommended where possible to reduce standardization, implementation, integration and validation efforts. Because SDN introduces a new structure of ecosystem with a larger number and more types of players, an increased focus on such models and mechanisms is required to facilitate interoperability. The Architecture specifically aims to supply a foundation for Information Model development.

1.4 Terminology

This section defines the terminology used in this document. Refer to the Architecture document [2] for a more complete list of Architecture related terms.

Term	Definition
API	Application Programming Interface: an Interface that can be directly called by software.
Application-Controller Plane Interface / Northbound Interface	An Interface permitting SDN Clients to interact with the network via SDN Controllers.
(SDN) Controller Plane	The architectural plane (layer) performing logically centralized control and management of network resources (grouped into Network Elements).
Data-Controller Plane Interface / Southbound Interface	An Interface through which a real or virtual Network Element can be controlled/configured/managed (e.g. OF-Config/Netconf), monitored (e.g. SNMP or sFlow), or have its forwarding behavior influenced by an external entity (e.g. by an SDN Controller via OF-Switch).

Term	Definition
Domain	A set of network resources (e.g. Network Elements) that reside in a common technology, address management, geographic, administrative, or asset ownership region, and that are controlled in some sense by a specific entity. An administrative domain is administered by the entity, a business domain is owned by the entity, etc.
Interface	A mechanism to permit software/hardware entities to interact, either within a processor/element (API) or between processors (RPC Interface / Protocol Interface).
Inter-SDN-Controller Interface	An interface permitting SDN Controllers to interact, either within the same administrative domain (intra-domain) or between administrative domains (inter-domain).
IP/IPv4/IPv6	Internet Protocol / IP version 4 / IP version 6.
MPLS/MPLS-TP	Multi-Protocol Label Switching / MPLS Transport Profile
Network Element	A collection of network resources that is managed as a unit. The Network Element terminates/originates traffic and/or forwards traffic. Examples include an OpenFlow switch, a web server, an L2 switch, an L3 router, a firewall, a load balancer, etc.
Network Function / Network Service	A specific data plane traffic forwarding/processing function (or set of functions) performed by the network, e.g. load balancing, network access control, content caching, etc. Some use the term Network Service for a set of associated individual Network Functions.
NFV	Network Functions Virtualization (refer to [1]).
OF-Config Protocol	The interface between an SDN Controller and a physical Network Element (physical switch).
OF-Switch Protocol (also known as OpenFlow Protocol)	The interface between an SDN Controller and a logical switch instantiated in a Network Element. Together with the OF-Config Protocol, this represents an instance of a Data-Controller Plane Interface (Southbound Interface).
OTN	Optical Transport Network.
Protocol Interface	An Interface implemented as a (network) protocol.
PIF	Protocol Independent Forwarding.
RPC Interface	Remote Procedure Call Interface, a type of Interface permitting interaction across a network.

Term	Definition
SDH	Synchronous Digital Hierarchy.
SDN	Software Defined Networking.
SDN Client	An entity interacting with the network (e.g. requesting resources or services) via an SDN Controller.
SDN Controlled	Controlled using SDN mechanisms (e.g. OF-Switch/ OF-Config, with network resources being controlled by the logically centralized SDN Controller).
SDN Controller	A software/hardware system performing logically centralized control and management of network elements while offering network services to zero or more tenants or applications. The SDN Controller may be physically implemented on a single platform, or it may be physically distributed.
SDO	Standards Development Organization.
SNMP	Simple Network Management Protocol.
Traditionally Controlled	Controlled using non-SDN mechanisms, typically using a distributed control plane protocol like Open Shortest Path First (OSPF) or Spanning Tree Protocol (STP), with these protocols typically being implemented on the Network Elements themselves.
Tunnel	A tunnel transparently conveys client traffic (i.e. traffic using the tunnel) from an origination point to a termination point over a server (underlay) network.
VNF	Virtual Network Function.

2. Illustrative Usage Scenarios

As the SDN Architecture is targeted at all network types, it needs to support a large variety of usage scenarios, for example the following:

- Automated provisioning of datacenters and various network layers (or combinations of network layers) to meet higher-level application / business objectives, for example considering virtual machine and/or application mobility as well as multi-tenancy.
- Enable diverse paradigms (e.g. centrally controlled) with respect to traffic management and traffic engineering for provisioning of various network layers (or combinations of layers) within a datacenter or across the WAN.
- Delivery of L4-L7 Services and other Virtual Network Functions (VNFs) by the controlled network itself, or by steering traffic to/through elements that implement these services, i.e. implementing traffic steering to/through a VNF Forwarding Graph (service chain).

Additional usage scenarios are listed in Appendix A.

3. Scope and Problems to Solve

The scope of the SDN Architecture in general needs to extend to the following. Note that the distinction between which items need to be addressed by the SDN Architecture vs. which items should be delegated to individual ONF working/discussion groups, projects or other SDOs may not be clear yet; where this distinction is already clear, it is mentioned here.

3.1 Overall Goal: Programmatic Abstracted Network Control

3.1.1 Logically Centralized Programmatic Control

The SDN Architecture needs to enable network and data-center operators, service providers, applications and application developers to *programmatically control* the network. This implies enabling applications to govern forwarding/processing of network traffic or manipulate network properties in order to achieve certain goals (e.g. improve network utilization, ensure traffic is forwarded to a specified set of destinations, implement the desired security or quality of service policies, etc.). For this to be possible in a meaningful way, applications also need access to network information pertinent to their functions; this information may be either static or dynamic, and may be provisioned, discovered, queried, or conveyed through notifications.

It should be noted that software has arguably been capable of controlling networks for decades. Programmatic control in SDN Architecture conformant SDN is decoupled and logically centralized while offering fine-grained control of network elements (e.g. forwarding based on many packet fields) as well as fine-grained and frequent feedback from network elements (enabling applications to become a part of a feedback loop, controlling the network and reacting based on the outcome). The nature of the software ecosystem has also expanded, enabling multiple types of applications (either performing distinct functions or acting on behalf of different tenants) to interact with the SDN Controller plane.

Furthermore, Software Defined Networking, as the term indicates, connotes software *defining* the network, i.e. starting with a raw resource and constructing a specific topology with specific behavior from it (by defining in detail how forwarding and processing should be performed) in order to achieve a specific objective / implement specific functionality. Software Defined Networking does not connote software merely *interfacing with* the network (e.g. configuring aspects of pre-existing functionality). In the OF-Switch variant of SDN, the SDN Controller controls switches in a fine-grained manner by downloading individual table entries, but other Southbound (from the controller) interfaces that operate at a higher level (e.g. the intent level) can also comply with this definition of SDN.

Figure 3 below depicts key aspects of logically centralized control: translating the high level intent into lower level instructions, and summarizing low level events and statistics to form higher level information.

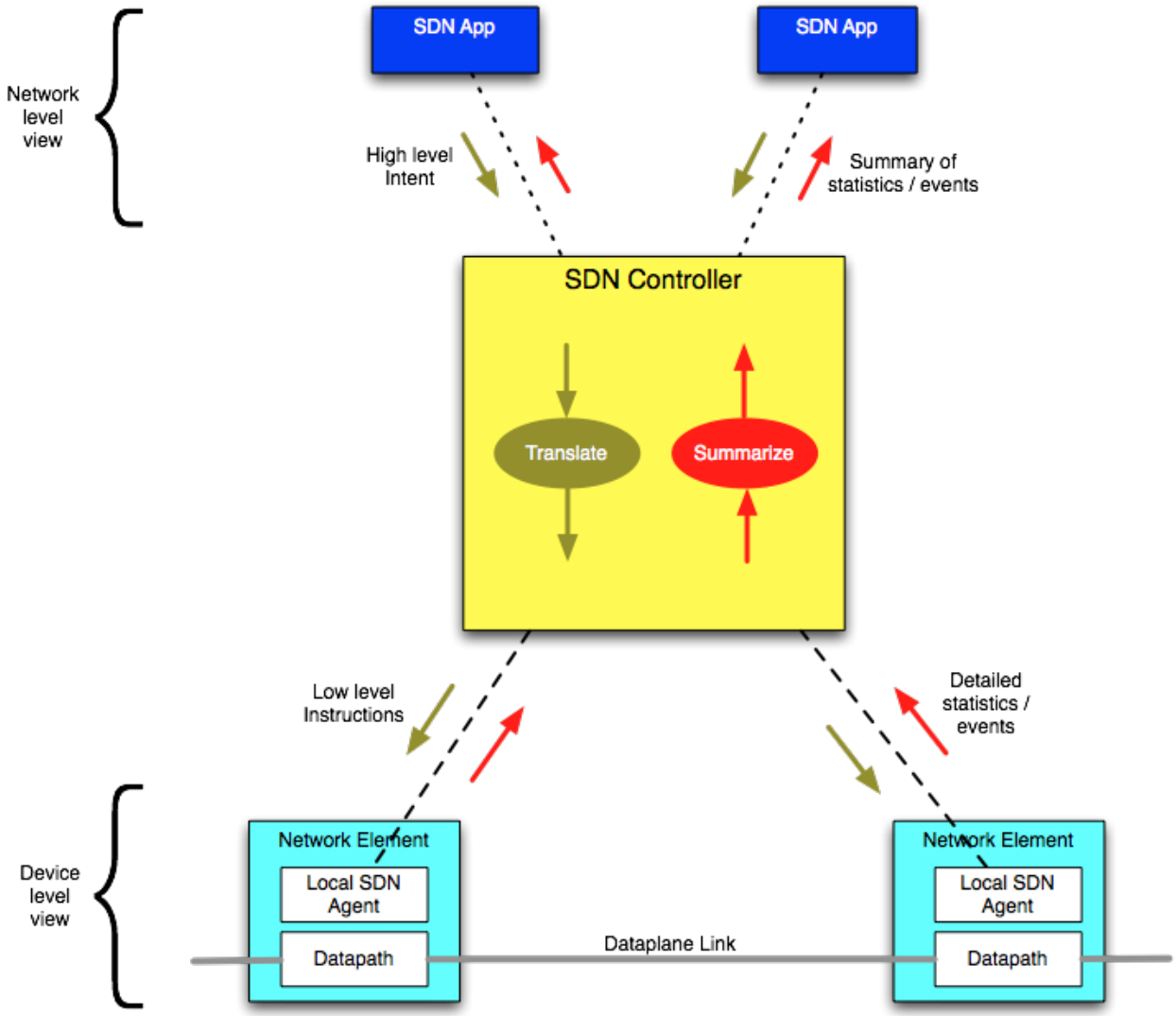

Figure 3: Logically Centralized Control and Information Translation

Figure 4 below illustrates that the interface southbound from the SDN Controller may be defined to be at a higher or lower level, and that it may be standardized by ONF or by other SDOs (or not be standardized at all), while still conforming to this definition of SDN.

Where a logically centralized SDN Controller plane does not exist at all, the network does not conform to this definition of SDN. This could for example occur when implementing control in a decentralized fashion by using distributed control/signaling plane protocols that permit the network elements to converge to the desired forwarding behavior (i.e. logical topology). We refer to this scenario as traditional control of the network.

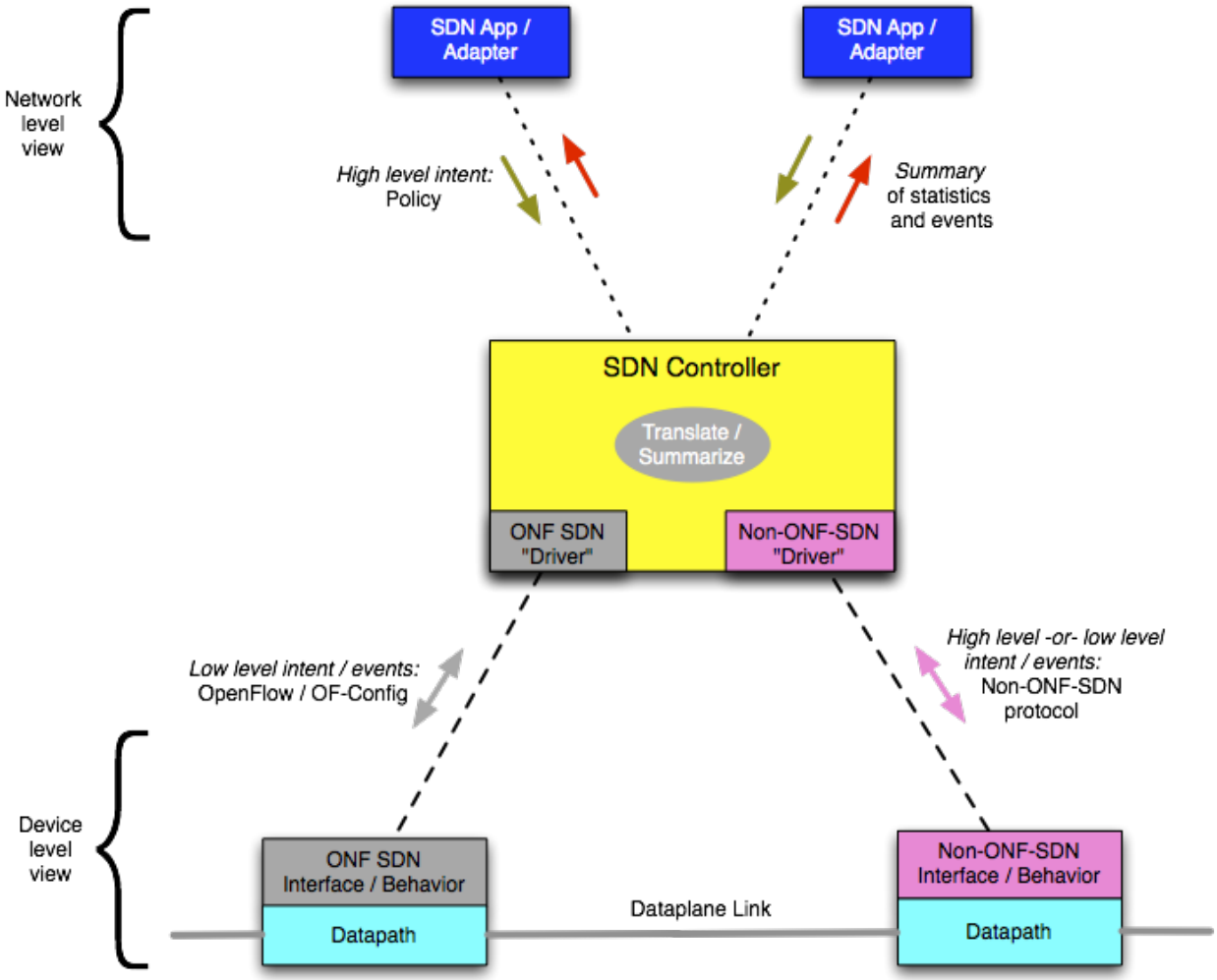

Figure 4: ONF vs. Non-ONF Southbound Protocols

3.1.2 Abstracted Programmatic Control

The controlling entities can operate on an *abstract* view of the network, where the level of detail exposed can vary according to the operation being performed. Details like network nodes and links may be abstracted and hidden.

Abstract representations of the network are implemented on underlying (real or virtual) network resources. This process is called *network virtualization*. The underlying resources assigned to a virtual network instance are sometimes called slices or partitions of the underlying shared resource. Virtualization can be implemented hierarchically, i.e. a virtual partition/slice can be further virtualized (thereby subdividing it at an additional level).

Whereas the term virtual network is usually employed to refer to a fairly complete representation of a network, an abstract view may only focus on specific aspects, e.g. connectivity, reliability, or security. Figure 5 below, for example, depicts an abstraction that merely records whether certain nodes are permitted to communicate with each other.

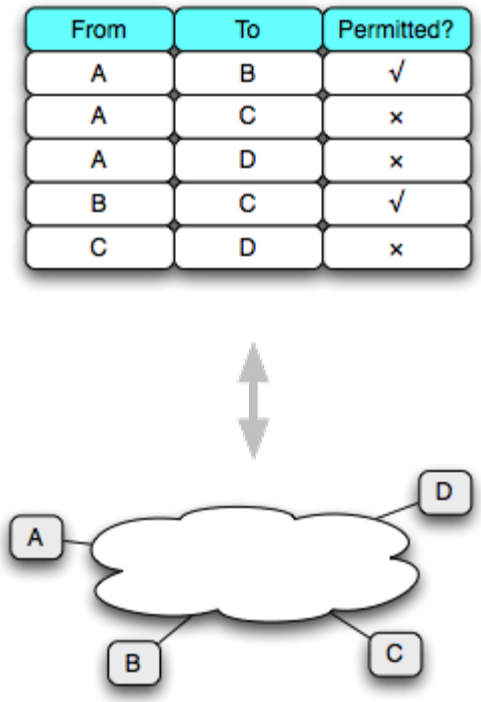

Figure 5: Is Connectivity Permitted Abstraction (Reachability Abstraction)

Network functions that are traditionally accommodated in discrete elements (e.g. load balancers or firewalls) may be implemented in converged elements that are managed as a single entity. This can be modeled as another form of abstraction of a network resource.

3.1.3 Components, Interfaces and Reference Points

The SDN Architecture needs to identify the entities involved in programmatic interaction with the network, including the most significant *components* and *actors* (e.g. applications, SDN Controllers and Network elements) and the *interfaces* between them. The actors may participate in the system in different ways, or have varying responsibilities, leading to the concept of distinct *roles* that could be associated with a given actor at different times. The Architecture should also define distinct *reference points* for significantly different points of interaction (e.g. distinct reference points should be defined for inter-domain and intra-domain interaction) as this will facilitate discussion and promote comprehensive coverage.

It should furthermore enumerate the standards that need to be created to support such interactions. Defining the standards themselves is beyond the scope of the Architecture.

3.1.4 Information Models

Information Models represent concepts, how these concepts relate to each other, as well as applicable rules and other constraints. The SDN Architecture needs to create a foundation for defining and developing Information Models. The SDN Architecture does not itself define all the required Information Models (nor representations thereof, e.g. Data Models and Interfaces), as this activity requires collaboration between multiple ONF Working Groups and SDOs.

Although different entities (e.g. network resources and functions in the SDN Controller) may be associated with different Information Model fragments, the elements common to these

fragments need to be consistent. Similarly, core elements may be media independent, with derivatives covering the media specific aspects.

3.2 Control Plane and SDN Controller Plane Technologies

3.2.1 Control Plane - Data Plane Decoupling

Traditionally, routers and switches run distributed protocols that respond to link-by-link network conditions. The SDN Architecture introduces the notions of decoupling the control plane from the data plane and logically centralizing the control plane. The term SDN Controller Plane is used to refer to a control plane with these characteristics.

3.2.2 SDN Controller Instance Distribution and Modularity

3.2.2.1 Distribution of SDN Controller Instances

The SDN Controller plane must scale to large networks (comprised of many nodes, links, users, applications and services). Provision needs to be made for assuring reliability. Both requirements may be supported by deploying multiple instances of SDN Controllers. These will need to communicate to distribute work, synchronize state, or otherwise coordinate cooperation, e.g. each SDN Controller instance may be responsible for a portion of the network. Distribution of control plane software components needs to be independent from the physical location of data plane nodes, except insofar as there are specific constraints (e.g. limits on communications latency).

Where a single logical SDN Controller is deployed on a physical distributed platform to increase scalability, the interface between its internal components need not be standardized. Where logically separate SDN Controller instances interact for other reasons, e.g. to span administrative or business domains, interaction should use a standardized protocol.

3.2.2.2 Modularity and Decomposition of SDN Controller Instances

Permitting SDN Controller instances to be modular would enable third parties to contribute algorithms that affect SDN Controller behavior. Each SDN Controller vendor can make arrangements with its partners to define appropriate APIs between the modules supplied by these partners and the SDN Controller itself. It remains to be seen whether and when the SDN Architecture and associated standards need to explicitly make provision for such decomposition.

3.2.3 Network Interaction Policies

SDN Controllers need to make provision for various entities to interact with the network (e.g. control it or manage it). Interactions need to be governed by policies that define which operations can be performed by each of these entities.

Mechanisms to express, distribute, and otherwise manage such interaction policies need to be created. Entities with the right to perform certain operations need to be able to delegate these rights to other entities. Policies would also govern the delegation process.

Although it is advisable to standardize interaction policy management/distribution mechanisms at some point, it is unclear when they need to be standardized. Usage models and the anticipated architecture will inform decisions with respect to what to standardize when. Note that once interaction policy related mechanisms are standardized, supporting infrastructure may also

need to be standardized, e.g. to permit entities to be identified and authenticated, and to maintain the privacy and integrity of interaction policy related data.

(Interaction policies themselves are ultimately governed by business decisions, and can therefore obviously not be standardized.)

3.2.4 Trust Domains

The SDN Architecture defines components entirely within particular trust domains. Interactions with other instances of trust domains are described via defined reference points. This enables the commercial and business interests of stakeholders to be reflected as abstraction barriers in the Architecture. Furthermore, the uniform treatment of this aspect is conducive to the design and audit of security measures.

3.2.5 Inter-Administrative-Domain Connectivity

3.2.5.1 Link on Boundary

Typically inter-administrative-domain connectivity is implemented by two Network Elements, each owned by a different entity, with the link between them straddling the boundary.

Initially existing protocols and mechanisms (e.g. Border Gateway Protocol - BGP) are expected to be used to facilitate connectivity between administrative domains over such links. In due course, equivalent mechanisms could be deployed between SDN Controllers to further simplify network elements and their operation. As the incremental cost of extending an existing mechanism may be small, the benefits associated with developing a new mechanism need to be significant to justify the considerable ensuing effort.

3.2.5.2 Network Element on Boundary

In the future, Network Elements might be deployed on the boundary (i.e. straddling the boundary) between administrative domains. In this case, several SDN Controllers could configure the Network Elements (e.g. with some ports assigned to each) and/or some ports could be configured and managed using traditional protocols and mechanisms (i.e. the element could be a hybrid element). Such shared control (custody) over a single Network Element could be implemented by having one SDN Controller have primary ownership of all resources, with the rights to manage/allocate some resources being explicitly delegated to another SDN Controller via a suitable protocol. Although an existing protocol may be deemed to suffice, it is likely that this will require a new protocol between SDN Controllers. This scenario is expected to be less common than the Network Element being at the boundary because of the difficulty of addressing concerns like joint property ownership, trust/security, joint physical access, etc.

3.2.6 Northbound Interfaces

The SDN Architecture provides for defining and optionally standardizing Northbound Interfaces (NBIs). Such interfaces provide for the generation of detailed or abstracted views of the network, both to present information to SDN Clients using the view, and to permit SDN Clients to configure/manage/control the network by interacting with the view. A detailed view can for example be presented where paths (including alternate paths) can be set for traffic, whereas with a further abstracted view, only the edge-to-edge service may be visible.

The SDN Architecture needs to make provision for multiple levels of abstraction (i.e. a hierarchy of abstracted views).

3.3 Network Element Forwarding and Processing Behavior

3.3.1 Remain Data Plane Agnostic

The SDN Architecture needs to accommodate circuit switched (e.g. OTN, SDH) and packet switched (Ethernet, MPLS/MPLS-TP, IPv4/IPv6) technologies. Provision needs to be made for interworking between technologies, e.g. between IPv4 and IPv6. As the Architecture is agnostic with respect to the employed data plane protocols, it can support predefined (e.g. standardized) as well as custom (user defined / dynamically defined) data plane protocols.

3.3.2 Support for Protocol Independent Forwarding and Datapath Programs

Protocol Independent Forwarding (PIF) enables the introduction of new data plane protocols. Header fields associated with these protocols are no longer predefined as PIF permits the parse tree to be configured. The match/action process can operate on these header fields, e.g. match them, copy them, increment/decrement them, etc., in order to implement the semantics required by the new data plane protocol.

Introducing support for new data plane protocols does not in itself require changes to the SDN Architecture as this is already data plane protocol agnostic, as detailed in the previous section. The PIF initiative, however, also introduces several new concepts that do need to be considered in the SDN Architecture:

1. The notion of a Datapath Program defining the behavior of the datapath, e.g. the arrangement of elements like parsing, matching, actions and traffic management.
2. The notion of modularity in these Programs, where a Datapath Program consists of modules (e.g. libraries) supplied by the main program's author or third parties (equipment vendors, operators, SDOs, or library vendors).
3. Toolchains to in effect compile and link Datapath Programs, from high level languages optionally to an Intermediate Representation, and eventually to whatever format suits the target platform.
4. New types of target platforms with diverse capabilities.
5. The notion of (program) configuration time, during which the Datapath Program is compiled and downloaded to the Network Element, in addition to the existing notion of run time (or operation time), during which the SDN Controller interacts with the Network Element.
6. New ecosystem players, e.g. new types of vendors and new standards bodies that produce and standardize the aforementioned components.
7. Lifecycle and deployment issues pertaining to these, e.g. dealing with upgrading a network to introduce support for a new protocol or a new version of a Datapath Program.

Some of these concepts are illustrated in Figure 6 below.

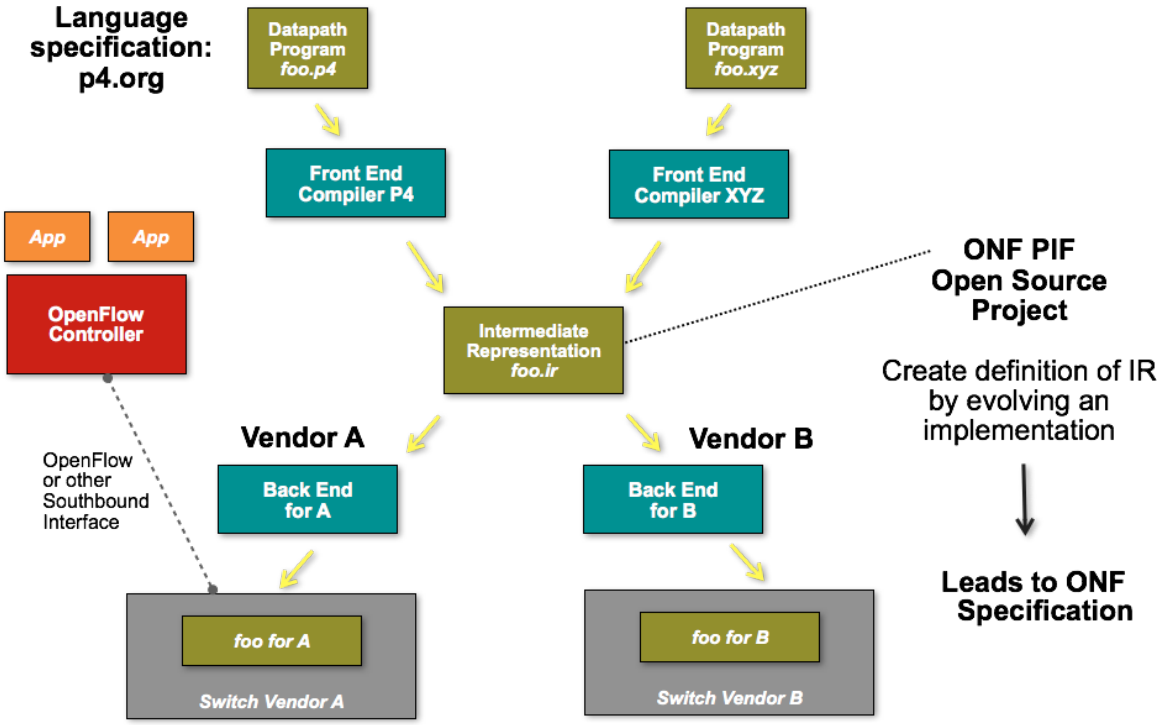

Figure 6: PIF Architectural Components and Standards Development Organizations

3.3.3 Support for Diverse Hardware / Software Platforms and Datapath Models

The capabilities offered by network elements vary widely with respect to media, port count, forwarding bandwidth, intelligence (e.g. classification and traffic manipulation capabilities) and statefulness (e.g. capabilities to perform stateful processing or capacities of the tables to maintain such state). Standards created by ONF need to take the various target hardware platforms into account by enabling multiple variants or subsets of standards (i.e. “profiles”) to be implemented on various types of hardware platforms. It is preferable for a single unified standard to be created that encompasses the subsets while permitting an appropriate subset of capabilities and associated parameters (e.g. performance/capacity/intelligence tradeoffs) to be detected and negotiated.

ONF needs to identify abstract (i.e. implementation-independent) models to represent forwarding and processing capabilities/behavior, and permit such models as well as associated parameters (e.g. available number of table entries, or available bandwidth that can be reserved) to be:

1. Created and optionally standardized (by ONF/vendors/others);
2. Negotiated between SDN Controllers and Network Elements;
3. Analyzed and transformed by tools (e.g. to predict performance or verify compliance);
4. Used for influencing forwarding/classification/processing behavior (by downloading table entries or parameters as in OF-Switch 1.x, or using more complex interaction mechanisms).

These models need to be able to represent fixed (predefined) protocol as well as variable protocol (i.e. protocol independent) Network Elements.

Note that as these models are intended to represent the behavior and capabilities at a high level, they do not duplicate the detailed expression of behavior encoded in the PIF Datapath Programs mentioned in the previous section. A model would be analogous to a ".h" file, which describes the capabilities of and the interface to a software application, whereas a datapath program itself would be a ".c" file, which contains the implementation itself.

3.4 Network Virtualization

This section considers the implications with respect to scope of network virtualization related requirements and usage scenarios.

Network virtualization is often employed to facilitate sharing of networking and compute resources between multiple users. Within a datacenter, for example, the virtual network instances would belong to datacenter tenants, and each virtual network instance would address the connectivity, security and addressing requirements of the real and/or virtual machines belonging to the tenant (or other entity) using that instance, while assuring traffic, management and control isolation of each tenant from all others.

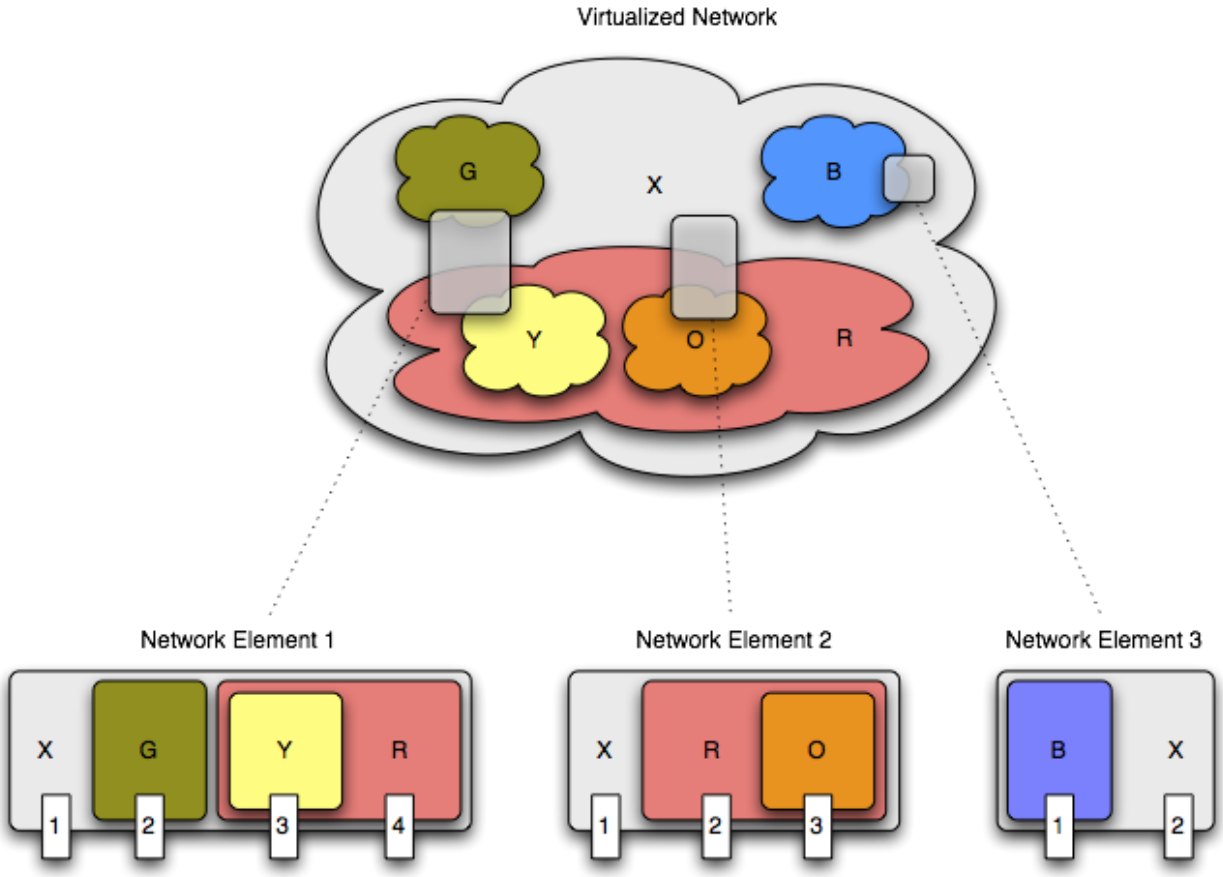

Figure 7: Virtualized Network and Virtualized Network Elements

Virtual network instances could themselves host multiple virtual networks. Figure 7 above depicts a scenario where the physical network X (grey in diagram) is partitioned into virtual networks R, G and B (depicted as red, green and blue respectively). The virtual network R is

partitioned again, yielding virtual networks Y and O (depicted as yellow and orange respectively).

The diagram also depicts the perspective of three individual network elements in this network. In the diagram, ports (i.e. interfaces) on the network elements are allocated exclusively to the underlying physical network or to one of the virtual networks. By employing mechanisms like tagging of packets or the establishment of tunnels, a physical port can also be shared among virtual networks, yielding the notion of a logical/virtual port, with multiple logical/virtual ports being hosted on a given physical port.

The following aspects need to be considered:

1. Creation of virtual network instances (by a management system), and provisioning of virtual network instances (by a management system or by the tenants themselves).
2. Interactions between general IT (e.g. compute/storage) control/management systems and the network, ensuring the network remains in sync as changes occur (e.g. to ensure that connectivity and ACLs move when VMs move).
3. Provisioning of tunnels (including specification of tunnel protocol parameters), relatively statically as well as potentially dynamically at high rates. When tunnels are dynamically provisioned, negotiation of tunnel protocols or protocol parameters may occur between the tunnel origination device and the tunnel termination device, e.g. IPsec cipher suites may be negotiated.
4. Support for stateful tunnels (e.g. tunnels entailing the use of sequence numbers and reordering or even fragmentation/reassembly) and Operations, Administration and Management (OA&M) for tunnels.
5. Tunnel specific behavior, for example the usage of multicast by Virtual eXtensible LAN (VXLAN) – in many cases, generic mechanisms will not be able to address all aspects of the required behavior.
6. Support for varying degrees of expressiveness with respect to deciding which traffic is forwarded using which tunnels.
7. Classification and differentiated processing of traffic contained in tunnels without terminating/originating these tunnels. Note that as this entails examining the headers within the tunnel, it may not be supported by all types of Network Elements. In cases where a tunnel instance only contains one category of traffic, it may be possible to determine the category and required processing based on the first tunneled packet, with the remaining packets associated with that tunnel instance inheriting the selected policy.

Other SDOs (e.g. the Internet Engineering Task Force - IETF) are expected to be responsible for the standardization of transport technologies and tunnel protocols. ONF would only take on standardization of such technologies/protocols if suitable technologies/protocols can't be identified or created by cooperating with other SDOs.

3.5 Supporting Network Functions Virtualization and Services

This section considers the implications with respect to scope of the need to support Network Functions Virtualization (NFV), L4-L7 Services, and other data plane Services.

3.5.1 Definitions

The NFV initiative [1] is being coordinated by the NFV European Telecommunications Standards Institute (ETSI) Industry Specification Group (ISG). NFV aims to use IT virtualization related technologies to enable deploying constituents of communication services on commercial

off the shelf servers (potentially with acceleration hardware installed to support greater port densities or higher throughput). The individual constituents are called Virtual Network Functions (VNFs). These are further subdivided into VNF Components (VNFCs). The VNFs are typically grouped and packaged/distributed/managed as Virtual Machines (VMs). Multiple VNFs are typically combined in a VNF Forwarding Graph to form a communication service. The composition process is called Service Chaining.

For the purposes of this section, an “L4-L7 Service” is defined as a data plane entity:

- Performing classification and/or differentiated forwarding (e.g. class of service / quality of service) and/or processing;
- Operating at L4-L7 (not merely at L2-L3);
- On a significant amount of data plane traffic (not just control/signaling plane traffic), precluding the hosting of such services on an SDN Controller.

As the name indicates, L4-L7 Services do not merely perform L2/L3 forwarding, e.g. they may perform complex classification or processing of traffic (involving higher level packet headers or packet content), stateful classification/processing, or other complex processing. This is typically implemented in software running on general purpose Central Processing Units (CPUs), with or without the assistance of processors or other acceleration hardware optimized for networking. As with NFV, L4-L7 Services can be packaged in VMs and chained.

The following sections use the generic term Service to refer to L4-L7 Services as well as other data plane services (not necessarily operating at L4 or above).

Due to the similarities between these sets of requirements / usage scenarios, they are treated together in this section.

3.5.2 Identifying Traffic Requiring VNF / Service Processing

Irrespective of where the actual processing related to VNFs/Services is hosted, Network Elements that deal with the forwarding of affected traffic need to be able to identify the traffic requiring such processing. ONF created standards need to make provision for various Network Element types (e.g. support stateful/stateless identification capabilities, small/large session tracking tables, shallow/deep inspection capabilities, etc.).

As the traffic identification/classification process may be incremental, the decision with respect to which processing is required may also evolve over time, e.g. as more packets are received, the identity of the user sending traffic or the application protocol may become known, with different policies (e.g. QoS) being associated with traffic forwarding/processing. The Network Elements involved in making services available need to be able to interact over the network to influence and coordinate this evolution. Note that as millions of interactions per second may be required to coordinate such processing at the microflow level for many concurrent microflows, routing all interactions related to coordination through SDN Controllers is typically not feasible, therefore the coordination will need to occur by Network Elements interacting directly with peer Network Elements, or by ensuring that the classification and processing operations are co-located within a Network Element (ideally within a single physical device).

3.5.3 Performing VNF / Service Specific Forwarding / Processing

Once traffic has been identified, VNF/Service specific forwarding or processing needs to occur. Where the forwarding/processing involves performing traffic management (e.g. implementing differentiated classes or assured quality of service / shaping etc.), or where low latency is

required, the VNF/Service will typically be hosted within the Network Element itself, but where the traffic is not particularly latency sensitive, processing may occur within the identifying Network Element or within some other Network Element, e.g. a VM in a data center. In either case, ONF standards need to permit the service to be invoked via appropriate actions and action parameters for services that are natively defined by ONF standards, or merely need to hand traffic to arbitrary software via a suitable local interface (i.e. API) or remote interface (typically protocol, optionally wrapped using an API) to provide for arbitrary services not (yet) standardized by the ONF. The mechanisms created to permit a single (local or remote) VNF or Service to be invoked need to be consistent with the mechanisms used to invoke a VNF Forwarding Graph or Service Chain (covered in the next section).

3.5.4 Service Chaining and VNF Forwarding Graphs

Where commercial services are assembled from a number of VNFs arranged in a so-called VNF Forwarding Graph, or where Services are combined from components, the notion of Service Chaining (Service Composition) arises. This entails diverting traffic through a series of services, arranged in a daisy chain or a more general topology. Although the chain could be hard-wired (therefore reflecting the actual physical network topology) for greater flexibility, traffic is often encapsulated in a tunneling protocol to permit conveying the traffic to an arbitrary destination (potentially by employing source routing) and to enable metadata to be conveyed. As such encapsulation / service function chaining protocols are (currently) not being standardized by ONF, and as several competing proposed protocols exist, the ONF SDN Architecture needs to be able to accommodate a variety of such protocols.

3.5.5 Single Vendor and Multi-Vendor VNFs/Services

Initially ONF defined standards will treat each VNF/Service as an opaque entity, without standardizing the subsystems present in the entity. This enables the use of a VNF/Service supplied by a single vendor and the use of a service supplied by a group of vendors, with the internal interfaces between subsystems in the VNF/Service in both cases initially not being standardized.

Once standards sufficiently address the deployment of such opaque VNFs/Services, ONF can consider standardizing the interfaces between subsystems within VNFs/Services to facilitate collaboration between multiple vendors in the creation of a single VNF/Service.

Furthermore, interfaces between VNFs and acceleration hardware should be defined, potentially by open source and/or specifications created by SDOs (not necessarily ONF).

3.5.6 Granularity of VNF Deployment and Usage

The SDN Architecture should consider the implications of the following specific deployment scenarios:

1. The use of SDN at a coarse grain, to direct traffic to a service chain (comprising various Network Elements including NFV platforms).
    - All traffic associated with the “gold” service class may for example be directed to the traffic cleaning service chain.
2. The use of SDN at a finer grain, within a platform hosting VNFs, to instantiate VNFs and steer traffic along the VNF forwarding graph.
    - The traffic cleaning service chain may for example be assembled by combining firewalling, intrusion detection, and malware detection VNFs.

- Furthermore however, traffic associated with a specific set of subscribers may be additionally directed to the parental control VNF, with other traffic omitting this step.

For both of these cases, but more critically for the latter case, the impact of VM migration (VNF migration) needs to be considered.

3.6 Traditional Networking Coexistence and Migration ("Hybrid")

The following scenarios and issues need to be addressed to ensure successful migration from / coexistence with traditional networking technologies. The sequence in which each scenario needs to be addressed will need to be determined.

Figure 8 below depicts several variants of hybrid networking. The following sections describe each variant in more detail.

Note that these variants are depicted as distinct cases in the interest of clarity. In reality, multiple variants can coexist, leading to a hybrid of hybrids, and resulting in a further increase in the complexity of Network Elements and the SDN Controllers tasked with controlling them.

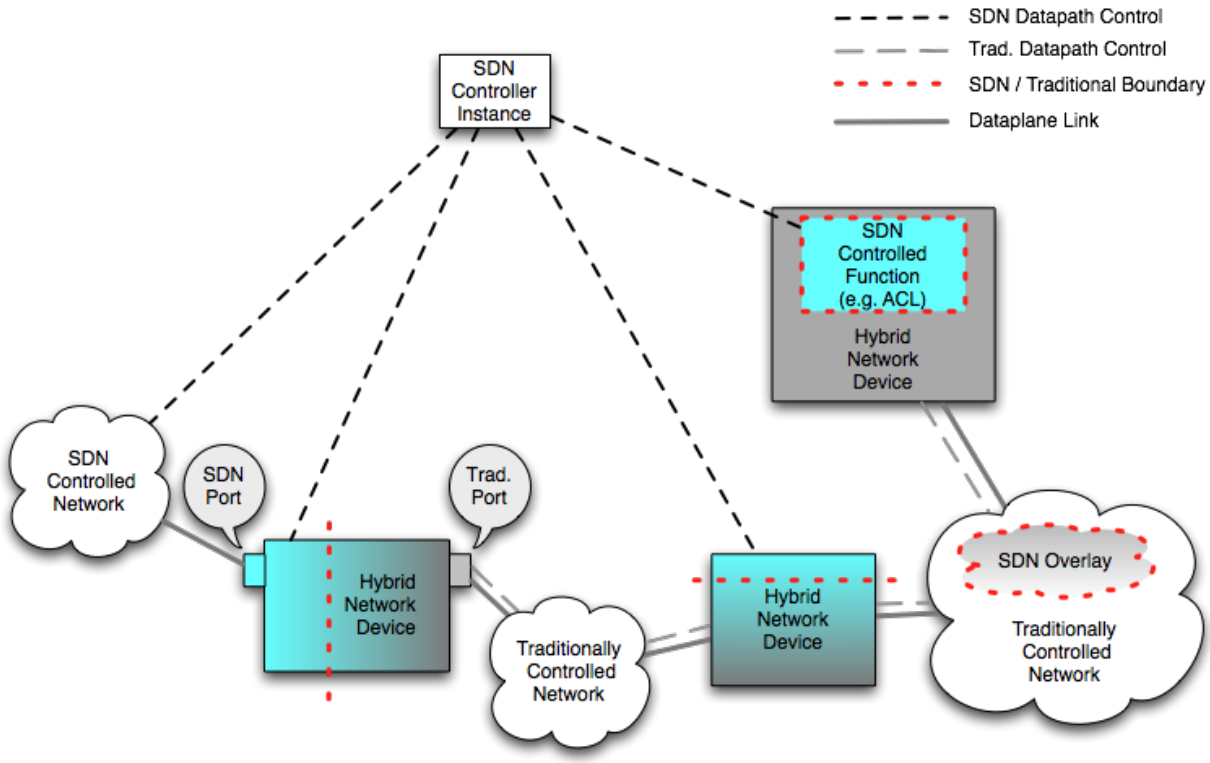

Figure 8: Hybrid Networking Variants

3.6.1 Traditionally Controlled and SDN Controlled Ports

The "Ships in the Night" concept in its strictest sense implies that Network Elements are partitioned into completely independent sub-elements, with for example one set of ports (physical interfaces) being assigned to a Traditionally Controlled Datapath whereas other ports (physical interfaces) are assigned to an SDN Controlled Datapath, as depicted in Figure 9

below. The concept can also apply to logical/virtual interfaces (i.e. specific VLANs on specific physical interfaces) or even individual flows. The SDN Architecture needs to define the mechanisms that are required to enable connectivity between the Traditionally Controlled and SDN Controlled portions of the Network Element.

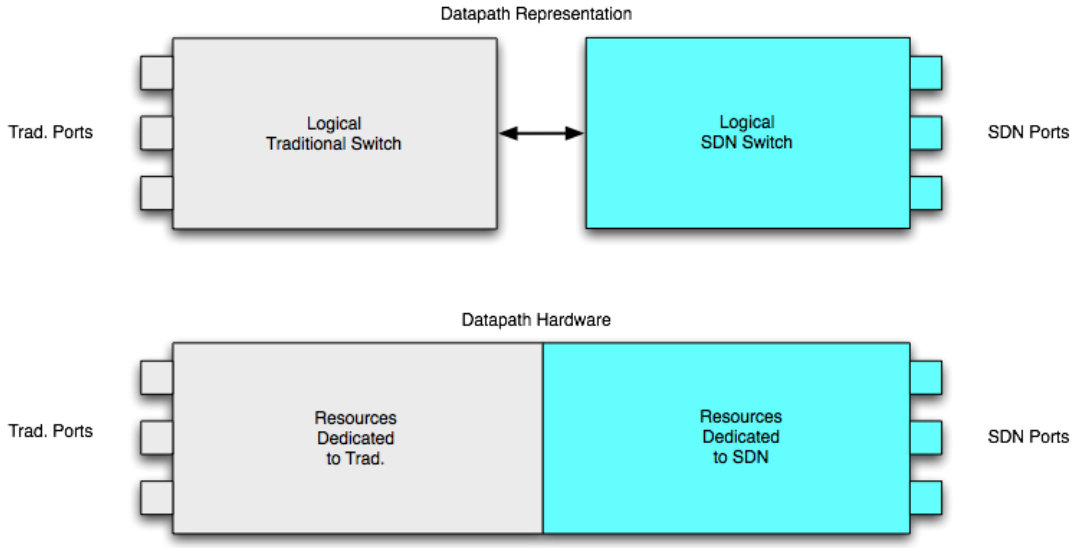

Figure 9: Port Oriented Hybrid with Dedicated Resources

There is also the case of “Ships in the Dusk”, where resources are not strictly partitioned between the traditional logical switch and the SDN logical switch – they are instead allocated from a common pool (refer to Figure 10 below). This violates the usual assumption that the SDN Controller has exclusive access to all of the Network Element’s resources that were made available to it. In order to cope with this situation, whoever attempts to allocate a resource needs to be prepared for the allocation attempt to fail.

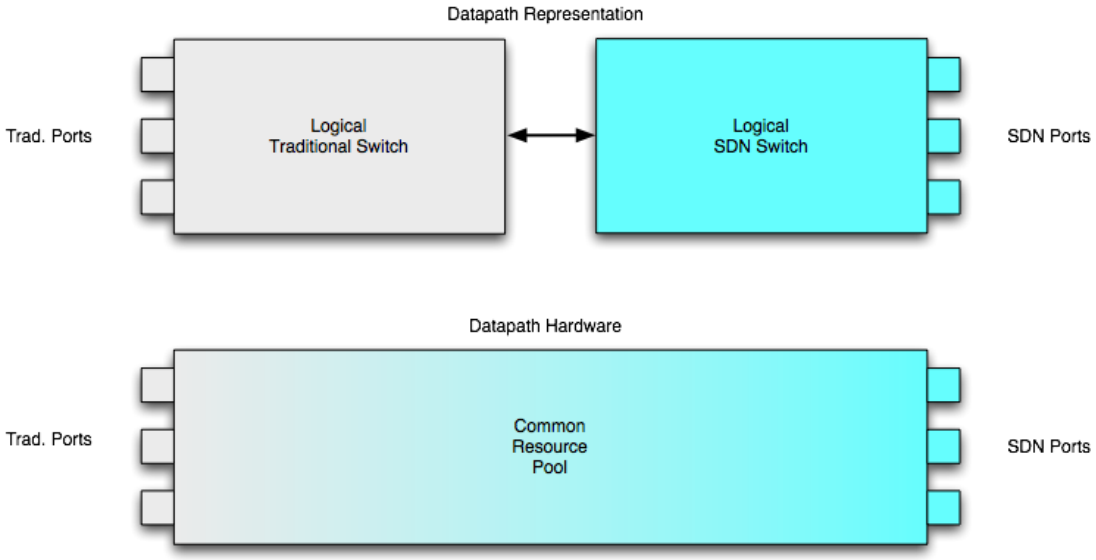

Figure 10: Port Oriented Hybrid with Common Resource Pool

3.6.2 Traditional "Underlay" with SDN "Overlay"

A network where forwarding is controlled using traditional mechanisms (e.g. where traditional L2 or L3 forwarding is used) can be used to carry the traffic for an SDN managed overlay network. Tunnels are established to carry the SDN managed traffic on the traditionally managed network.

In some cases, traffic needs to be encapsulated to transit the traditional network, leading to the concept of overlay on and off ramps (performing tunnel origination/termination, i.e. encapsulation/decapsulation), as depicted in Figure 11 below.

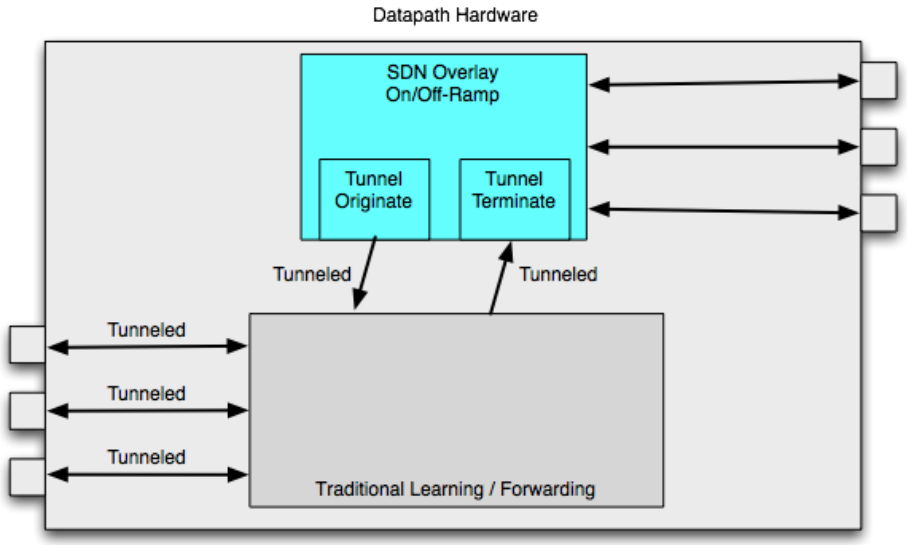

Figure 11: Overlay On/Off-Ramp Gateway

It is also possible for the underlay network to be controlled via SDN, while the overlay is traditional.

A variant of this case occurs where virtual switches are used. In this case, the virtual switch can act as the combination of a tunnel termination/origination point, a traditionally controlled switch or endpoint, and an SDN controlled switch. The entities attached to the virtual switch may be virtual machines that contain SDN controlled endpoints (virtual network interface cards).

3.6.3 Partitioning of Responsibilities

In this case, SDN (e.g. OF-Switch compliant) forwarding/monitoring/processing coexists with traditional traffic forwarding (e.g. L2/L3 forwarding) or even more complicated traditional networking functions that involve traffic processing (e.g. IPsec termination/origination). Here the SDN managed classification operations and actions are used to implement value added processing (e.g. classification into categories for QoS purposes or firewalling) while traditional mechanisms continue to be used for forwarding. Figure 12 below depicts such a scenario.

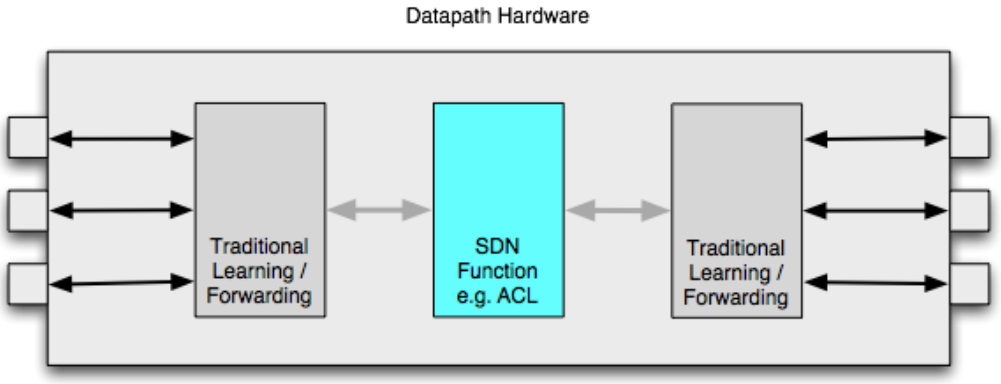

Figure 12: SDN Controlled Function in Traditionally Controlled Datapath

The value added processing can also be performed by higher-level applications/services, e.g. L4-L7 or complex/content aware security services. Some network operators would prefer to standardize on automated centralized provisioning/management/monitoring of their networks, and would therefore want a single system to be able to perform these tasks for all logical/ physical interfaces / network elements, irrespective of whether traffic forwarding is signaled/ controlled via SDN or traditional means for these entities. Unambiguous and strict partitioning of responsibilities, for example by permitting forwarding to be controlled via SDN whereas statistics are queried via traditional protocols, or enabling forwarding to be controlled via traditional protocols while security (allow/discard) is controlled via SDN, will facilitate implementing this scenario.

In the most general case one would permit all aspects to be simultaneously configured, monitored and in general managed using traditional and SDN mechanisms, with changes implemented via one mechanism being reflected via notifications to other mechanisms. Due to the technical difficulty of supporting this scenario, standardizing it is currently not envisaged.

3.6.4 Explicit Invocation of Traditional Forwarding from SDN (NORMAL Reserved Port)

Another scenario involves SDN performing the primary traffic classification and processing function (including traffic forwarding, should the flow entries so dictate), but with the option of SDN explicitly delegating forwarding to a traditional forwarding element. This is encoded in OF-Switch by directing traffic to the NORMAL reserved port. The scenario is depicted in Figure 13 below.

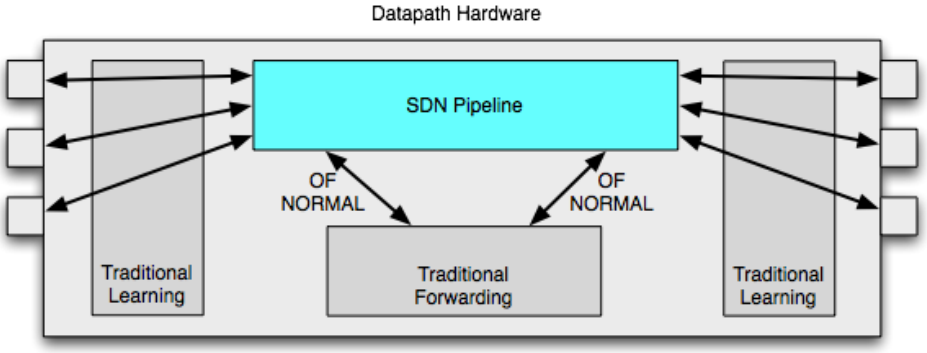

Figure 13: SDN Explicitly Invoking Tradition Forwarding

3.6.5 Traditional/SDN Control Plane Connectivity – Intra-administrative-domain and Inter-administrative-domain

In order to avoid having to statically configure forwarding (e.g. supply static routes for L3 or manually prevent L2 loops), the SDN Architecture needs to make provision for control plane connectivity between SDN controlled and traditionally controlled networks. Such connectivity would typically use the media attached to the Network Element. The control plane itself (e.g. routing protocol handling software) could be embedded in the Network Element, or the control plane traffic (e.g. routing protocols) could be forwarded to the SDN Controller for processing. The latter case is depicted in Figure 14 below. The appropriate protocols for the specific scenario (e.g. OSPF / IS-IS / BGP etc.) would need to be supported.

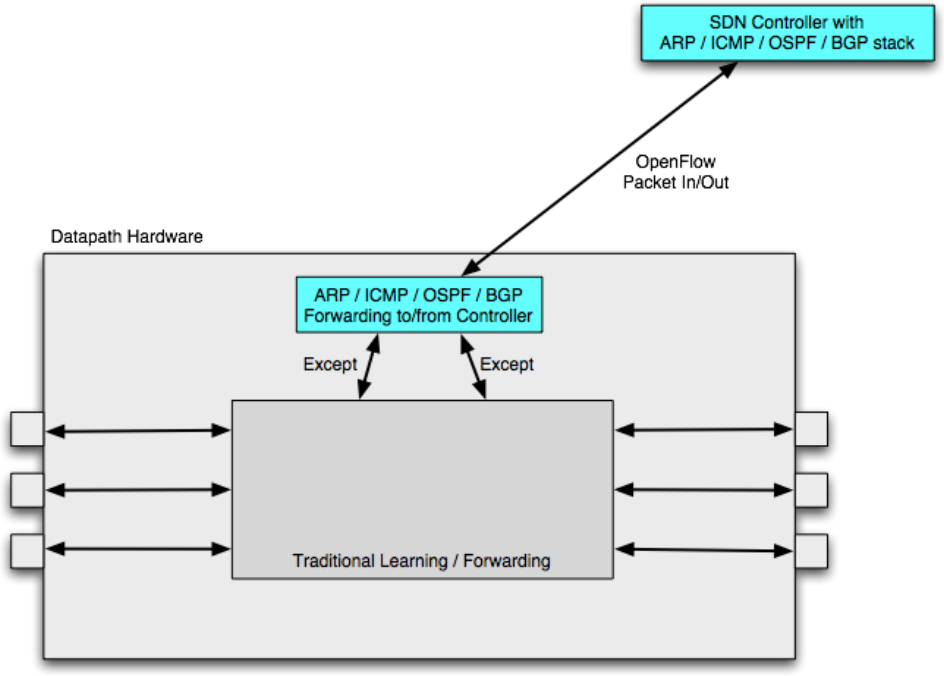

Figure 14: Traditional Control Plane On SDN Controller

4. Requirements

The SDN Architecture needs to address the following requirements.

4.1 Simplicity and Expressiveness

Generic semantic models that can represent multiple abstraction layers need to be identified. Such semantic models can include specific Information Models, but models that encompass behavior, e.g. more general domain models, also need to be considered.

The Architecture needs to make provision for introducing additional layers of abstraction in the future.

The SDN Architecture should minimize the surface area of the standards and architectural components while maximizing the generality thereof.

4.2 Applicability

4.2.1 Definition of Scope and Value Proposition

The SDN Framework and Architecture need to clearly, precisely and concisely define the scope and value proposition for SDN.

The SDN Architecture needs to ensure compatibility with existing network paradigms, standards, and common practices. Packet and circuit switched networks need to both be supported. Emerging new paradigms, for example flow-based packet forwarding, that take many header fields into account (i.e. forwarding that is not merely based on the destination address and labels/tags), also need to be supported. The various paradigms need to be able to coexist. Abstractions and standard protocols need to be defined to allow for rapid adoption of SDN solutions into existing operational environments motivated by the most business critical usage scenarios.

4.2.2 Network Types and Paradigms

The SDN Architecture needs to make provision for supporting all types of networks, including wide-area transport networks, transport services, data center networks, intra-site service chaining (including VNF chains) and residential IP services. The Architecture needs to state whether a one-size-fits-all approach is sufficient to control the in-scope types of networks or whether variations of the basic SDN scheme are required and if so, to what extent.

4.2.3 Evolution over Time

Tradeoffs will need to be made between time to market and quality/breadth of implementations. Although the SDN Framework and Architecture are not intended to constitute a roadmap or a work plan, they can and should identify which problems/requirements have higher priority/urgency than other problems/requirements, enabling ONF's working groups to address the highest priority/urgency items first.

The SDN Architecture should focus on fully supporting specific identified usage scenarios instead of poorly supporting fractions of a larger set. Additional usage scenarios can be supported at a later date.

4.3 Interworking

Deployability of SDN is facilitated by remaining compatible with existing technologies where possible. Where possible, supporting hybrid usage scenarios via guidelines and best practices instead of creating additional standards or variants of standards is advisable in order to reduce the effort expended on standards creation.

4.4 Interoperability

As interoperability is the primary goal of most standards, ensuring interoperability needs to be a key focus item.

The SDN Architecture can and should promote interoperability by ensuring elegance, clarity and completeness of coverage in standards, thereby also facilitating more complete/correct/secure implementations.

Standardizing a single approach / information model / data model / interface facilitates interoperability. Where these cannot be standardized, best practices can at least be documented. The SDN Architecture itself does not specify interfaces in detail or document such best practices, as this is the remit of individual working groups / projects within ONF.

4.5 Scalability

The following concerns apply with respect to scalability requirements:

1. Scalability of the SDN Controller plane:

   Although logically centralized, a single SDN Controller instance can be implemented as a distributed system, spanning multiple physical platforms, in order to improve scalability and availability. As this is an implementation-specific concern not visible to entities interacting with the SDN Controller, it is not discussed further in the Architecture. Furthermore, the SDN Controller plane can contain multiple SDN Controller instances, arranged in a tree or graph, with each level presenting a virtual view of the network to the higher level, thereby further increasing scalability.

2. Scalability within a Network Element:
   A Network Element may be implemented as a distributed system, potentially spanning a heterogeneous set of processors/ASICs, with internal load balancing / cascading / clustering to achieve performance (throughput/latency) and capacity goals. If the distributed system is represented as a single network element to the SDN Controller, this may not be visible to the SDN Controller, and the interfaces within the distributed system may not be standardized. If the distributed system is comprised of multiple Network Elements visible to the SDN Controller, with the combination being exposed as a single virtual Network Element, the existence of the distributed system does need to be considered by the Architecture and by standards.

3. Scalability specific to individual functions/capabilities:

   Which functions are implemented in a fast vs. a slow path may differ for various elements, and the supported capacities may differ, therefore standards need to make provision for a wide range of capabilities/footprints and permit negotiation of feature sets and associated parameters at initial configuration time or even at arbitrary times.

4.6 Security

When security is discussed in the context of SDN, it can refer to the following:

1. The use of SDN to implement network security;
2. The security of SDN infrastructure and mechanisms itself.

Security concerns pertaining to the SDN infrastructure itself (e.g. SDN Controller plane traffic) can be grouped into the following categories:

1. Authentication of communicating entities (e.g. SDN Controllers and Network Elements);l
2. Access Control by enforcing configured policies that govern rights for each entity;
3. Privacy by encrypting communication channels using protocols like (D)TLS/IPsec;
4. Auditing by maintaining and permitting access to applicable records, for example records that capture the identity of the initiator and/or the details of an operation;

5. Certification of Assurance, e.g. via a Common Criteria evaluation lab;
6. Denial of Service Mitigation by policing (metering).

The Architecture needs to make provision for the mechanisms without mandating which mechanisms are deployed when/where. The SDN usage environment will in the end determine which of these mechanisms are actually required and deployed. The value added by the Architecture is to clearly identify reference points at which security considerations apply, at whatever level of detail/stringency suits the particular needs of an implementation.

Although similar concerns exist for data plane traffic, i.e. where SDN is used to implement network security, such concerns are deemed beyond the scope of the SDN Architecture. Ensuring that the L4-L7/Security Service and VNF Chaining usage scenarios can be supported is deemed to suffice.

4.7 Resilience and Fault Tolerance

4.7.1 Error Handling

The SDN Architecture needs to define in general terms how error conditions need to be handled. Error conditions should be handled as gracefully as possible, with erroneous inputs on any Interface or malformed network traffic being counted and (if configured, subject to rate limiting) reported, and without such inputs affecting unrelated subsystems or unrelated state (e.g. erroneous input should not result in unintended configurations being implemented or security vulnerabilities being introduced).

4.7.2 Transient Condition Handling

The timescales of changes implemented by SDN Controllers vary. In some cases, mechanisms like a multi phase commit process and make before break need to be employed to ensure transients are minimized, and/or an interim configuration needs to be deployed to ensure that undesired transients (e.g. security issues) are eliminated. The standards need to make provision for the global application of consistent changes at both the network structure level (e.g. instantiation of virtual network elements) and the flow level (e.g. updates to flow tables).

4.7.3 High Availability Support

The SDN Architecture needs to make provision for implementing highly available networks. This implies supporting redundancy of links and Network Elements (i.e. for the data plane) and of SDN Controllers (i.e. for the SDN Controller plane). Redundancy of platforms could imply state synchronization and state sharing (either in an active/active, i.e. load sharing, or active/passive, i.e. hot standby configuration).

4.8 Management and Monitoring

The SDN Architecture needs to take into account the management of the SDN system itself, for example by considering arrangements for pairing Network Elements and SDN Controllers as well as by making provision for performance monitoring of the SDN Controller/Network Element interfaces.

Further concerns include configuration, bootstrapping, and in-band control of individual Network Elements while addressing the buildout, repair and maintenance of networks.

The SDN Architecture shall support fault and performance management. Troubleshooting and debugging shall be supported. In order to enable monitoring of network state, traffic, SLAs and network performance in general, the Data-Controller Plane Interface (Southbound Interface) and the Northbound Interface shall support the flow of relevant information from the Network Element to the SDN Controller in addition to the configuration/programmatic information flowing in the converse direction. Care should be taken to select the granularity and abstraction level of information to both enable scaling and provide enough useful information.

Depending on the specific aspect, existing standards may suffice, new standards may need to be defined, or it may be expedient to employ best practices and guidelines. Where management infrastructure is already available and has already been standardized, the SDN Architecture needs to take the semantics of such infrastructure into account, only making changes to such infrastructure (e.g. redefining it to use SDN mechanisms) where required.

- Changes may be required to accommodate bootstrapping SDN without a traditional network being present. (Whether this is required is controversial.)
- An area where changes may conversely not be required is the installation and upgrading of software.

As detailed investigation may be required to determine whether or not existing mechanisms suffice, the SDN Architecture does not supply guidance for each individual area or mechanism.

5. Conclusions

The SDN Framework document delineates the scope of the SDN Architecture, including the requirements that it needs to address and the problems that it needs to solve. As the extent of the implied work is significantly larger than can be addressed in the first version of the SDN Architecture document, several versions of the Architecture document will need to be created to fully address the delineated area. When selecting which items should be covered in the first version, the appropriate balance between simplicity and expressiveness as well as between comprehensiveness of coverage and time to release the document will need to be found. Preparation of the Architecture document as well as other insights that emerge over time are conversely expected to result in further revisions of the Framework document. Participation in the process to evolve the SDN Framework and Architecture documents is welcomed.

6. References

1. Network Functions Virtualization (NFV) white papers and other documents: http://www.etsi.org/technologies-clusters/technologies/nfv
2. SDN Architecture v1.0: https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/TR_SDN_ARCH_1.0_06062014.pdf
3. SDN Architecture Overview v1.1: https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/TR_SDN-ARCH-Overview-1.1-11112014.02.pdf

7. List of Contributors

The following individuals contributed to this document: Bernd Zeuner, Beth Most, Dan Malek, Daniel King, Dave Hood, Dave McDysan, Dave Meyer, Eve Varma, Deepak Bansal, Fabian Schneider, Johann Tönsing, Malcolm Betts, Manuel Paul, Nabil Damouny, Reda Habib, Rob Sherwood, Sibylle Schaller, Stu Bailey, and Zoltán Lajos Kis. (In many cases they contributed content on behalf of their organizations.)

8. Revision History

Date	Rev	Description	Editor
2015-07-22	1.0	Initial public version.	J H Tönsing

Appendix A: Usage Scenarios

The SDN Architecture is expected to support a wide variety of usage scenarios. In order to enable gaps and deficiencies in the Architecture to be identified, the extent to which the Architecture supports the key usage scenarios listed in Section 2 needs to be determined.

The following additional usage scenarios can be considered when time permits.

Carrier

1. Edge switch for service aware routing: QoS
2. VNF instantiation, orchestration, management and control
3. Carriers wholesaling + retailing capacity
4. Inter-datacenter WAN network provisioning for carriers
5. Multilevel provisioning by carriers (optical / MPLS / IP etc.)
6. Multi-administration operation in federated or hierarchical modes
7. CDN traffic management
8. CDN cache management
9. Additional requirements / scenarios:
    a. centralized, automated control and provisioning model
    b. support multi-tenancy
    c. support establishment, monitoring and maintenance of SLAs based on parameters including bandwidth, latency, jitter, drop
    d. allow optimization of network resource usage against a wide range of criteria (customer utility, operational and capital costs, energy usage, particular resource bottlenecks) using multiple utility functions
    e. increase service velocity and increase the reach of applications by providing good network abstractions
    f. support for integration of telecommunication technologies and application of SDN principles to Ethernet, MPLS and optical transport networks

Enterprise and Campus

1. Requirements / scenarios:
    a. centralized, automated control and provisioning model to support the convergence of data, voice, and video as well as anytime, anywhere access;
    b. enforce policies consistently across both the wired (e.g. Ethernet) and wireless (e.g. WiFi) infrastructures;
    c. automated provisioning and management of network resources, determined by individual user profiles and application requirements, to ensure an optimal user experience within the enterprise's constraints.

Datacenter and Cloud

1. Cloud / datacenter network virtualization
2. Automated VM + app migration
3. Integration with storage
4. Rapid provisioning of cloud services
5. Stringent security requirements for virtual networks
6. App influencing network / network catering for app's demands
7. Big Data acceleration – reconfigure network per job or phase in job
8. Hyper-scalability
9. Elastic allocation of network resources
10. App aware of network, can influence network to meet app's needs
11. WAN/LAN Optimization
12. Inter-datacenter WAN with centralized traffic engineering
13. Bandwidth optimization
14. Hybrid clouds
15. Public - private cloud (spill over, encrypted tunnels)
16. Flexible hand-off to the external cloud provider

Common to Various Network Types

1. Carrier - enterprise boundary (shared control / management)
2. L4-L7 + security services, fine-grained security, avoid inserting specialized elements in path of all traffic
3. Usage scenarios for each design point – overlay / underlay / middlelay
4. Ecosystems - consumers and producers
5. Simplified OA&M for large networks
6. Multi-domain e.g., to enable multiple controller vendors, connect geographic domains or implement partitioning to provide scaling (e.g., in terms of response time)
7. Resilient, efficient interconnection of an OF-controlled domain with a legacy bridged/routed domain
8. Enable and not restrict innovative controller-based deployments to realize new or enhanced services

