Software Defined Networking, DOCSIS Provisioning, and MSO Commercial Services

Kevin A. Noll
Time Warner Cable

Wesley George
Time Warner Cable

Richard Loveland
Alcatel-Lucent

Abstract

In recent years, cable operators have realized significant growth outside the traditional residential market. The North American cable industry is set to top $8.5 billion in revenue from offering network services to commercial and carrier customers. In a $130 billion telecom services market, cable operators have plenty of opportunities to continue growing this segment of their business. However, to better compete and realize revenue faster, cable operators need to offer new and innovative services, move them to market faster, simplify operations and significantly increase their rate of customer turn-up while decreasing the time to delivery.

This paper will examine how Software Defined Networking concepts and the Cablelabs DOCSIS, L2VPN, and DPoE specifications can be integrated to enable cable operators to realize fully automated end-to-end provisioning of commercial services and dynamic network reconfiguration through a single and simplified provisioning interface.

INTRODUCTION

In the mid to late 1990's Internet access was slow or expensive (usually both) and dominated by the telephone companies and dial-up connections. The cable industry was investing heavily in two-way plant infrastructure for interaction between customers and services. Much of this investment was dominated by Hybrid Fiber Coax (HFC) builds. This investment was the stepping-stone to allow the cable operators to push into the always-on high-speed Internet access service. By utilizing the two-way capabilities of HFC, the cable operators were able to economically begin to offer residential high-speed Internet access at affordable prices.

In order to take advantage of the HFC investment for high-speed Internet access, the cable operators defined a protocol called Data Over Cable Services Interface Specification (DOCSIS) of which the first version was released in 1997. This specification allowed vendors to create cable modem systems that would allow users to access the Internet and its growing collection of content in a more user friendly and faster method.

The introduction of cable modem systems and the tremendous growth of users demanded a set of provisioning and operational tools to support large scale transactions for automated provisioning and troubleshooting. These tools developed over time into an operational support system that is used by cable operators around the world to manage cable modem services with thousands of transactions a day.

As Cable service providers expand into serving enterprise as well as residential customers with fiber services, that same scale and transactional operation model is needed. Debate is ongoing within the MSO community whether a new OSS system should be developed for these new network models, but the prevailing direction has been to augment fiber access systems so they can be provisioned in a nearly identical manner allowing the reuse of massive investments in DOCSIS OSS systems that have served the industry well.

As a result in 2009 the operators, along with Cable Labs, began to define a set of specifications called DOCSIS Provisioning of EPON (DPoE) to specify how an EPON fiber access system should be provisioned. These specifications leverage the investment in cable modem operational support systems emulating the provisioning of a cable modem (CM) and Cable Modem Termination System (CMTS).

As part of this emulation, the Optical Line Terminal (OLT) Emulates a CMTS and creates a software instance of a cable modem called a virtual Cable Modem (vCM). Network Function Virtualization (NFV), which the vCM is a form of, has become one of the latest driving trends in the telecommunications industry. The goal of NFV is to reduce the cost of devices by relocating much of the intelligence for network devices, especially in the home, up into the network. This keeps the physical hardware as simple as possible with a standardized applications programming interface (API) to set up the hardware device – in this case an Optical Network Unit (ONU). DPoE provides an API to the ONU called DPoE OAM. In addition the vCM is provisioned through the standard CM configuration file allowing the provisioning of the vCM to be an abstracted view of what of the service being offered to the user. Reuse of the CM provisioning and operating model provides a standard method of defining the service while it reduces the CAPEX needed for OSS development and allows automated provisioning of these fiber systems. This helps reduce the time-to-market of the volume rollout of EPON. Of course EPON could be used on a much smaller scale with vendor provided EMS applications but those systems do not make a large scale deployment possible in the timeframe desired.

From the network perspective, DPoE enables an endpoint (vCM) to be provisioned to utilize network protocols such as MPLS and BGP signaling to automatically set up connections across the network to the far end of the MPLS tunnel creating a layer-2 virtual private network (L2VPN). This ability follows the desire of another trend in the network called Software Defined Networking (SDN). This ability to set up these network wide services can dramatically improve service delivery velocity increasing the number of transactions per day and greatly reduce errors in setting up complex network connections. The combination of DPoE and SDN concepts creates a powerful tool for the cable operators.

This paper is intended to examine the commonalities and benefits of combining DOCSIS-based provisioning with SDN and NFV to show how the cable operators can benefit from implementing such an architecture for its customers, It will discuss how these concepts create L2VPN services in a highly automated method.

SOFTWARE DEFINED NETWORKS

Software Defined Networking (SDN) is an increasingly common buzzword, especially when discussing the interaction between applications, networking, and the infrastructure they share to provide services “in the cloud.” Unfortunately, SDN is an overloaded and broad term that if one were to ask three different “SDN Experts” what SDN is, it would likely result in five different answers.

Typically, a discussion of SDN will start with separation of the network's control plane from the forwarding plane, decoupling forwarding and policy decisions (what traffic goes where) from network transport and topology (how the devices are interconnected). This is the focus of projects like the Open Networking Foundation's OpenFlow. This project is defining a set of specifications allowing software (the SDN controller) running on a general purpose computing platform to perform the control-plane computations and functions required to dictate the flow of data through the network. The SDN controller uses OpenFlow to directly manipulate the forwarding tables of the switches and routers in the network. It is important to stress that SDN and Openflow are not interchangeable terms, and a network can be Software Defined without the use of Openflow.

Many service providers operate their network with little or weak central control over its configuration and management. This means that the network configuration and state is effectively stored in a giant distributed database. This is not inherently a bad state of affairs, but network operators aren't always good at getting the information in that giant database into a form that is usable for making business decisions that optimize the use of the network and the services that run over it.

Separation of control and forwarding enables programmatic orchestration and provisioning of network resources and optimal, secure flow between application services. It should be clear that consolidated control should lead to provisioning activities that are consolidated through a single programmatic interface located at the controller. It should also follow that centralizing the flow of network metrics would reduce the complexity of dealing with that giant distributed database.

With consolidated provisioning flow it becomes necessary to create a well-defined and standardized API through which the business can interact with the controller. Ideally, this API would create a layer of abstraction between the business systems and the network. Through abstraction and standardization, the network operator is able to reduce the time to develop and deploy new services. The operator is able to reduce (if not eliminate) human-induced network configuration errors.

Especially in the context of complex Commercial Business offerings, the operator is able to reduce the service provisioning process to a transaction rather than a complex series of steps involving multiple systems and humans.

In short, SDN is a method to enable a high degree of automation in provisioning and managing network services. While this can be done by directly manipulating forwarding tables as with OpenFlow, it can also be done by leveraging existing control plane and configuration methods such that some amount of traditional, distributed network control plane remains in place. Automation in this context is primarily focused on improving efficiency, speed, and accuracy. While there are certainly places where automation can eliminate time-consuming and repetitive manual processes, automation does not have to be hugely complex logic that expects the network to make most decisions via autonomic intelligence without human involvement. The goal isn't to build Skynet (of the Terminator franchise), because the tools and expertise necessary to build that level of independent, intelligent network are fairly limited. Instead, it can be a set of pre-defined scenarios that a human selects and executes, such that it serves as a force multiplier to enable the smart people running the network to do more, do it faster, and do it more accurately.

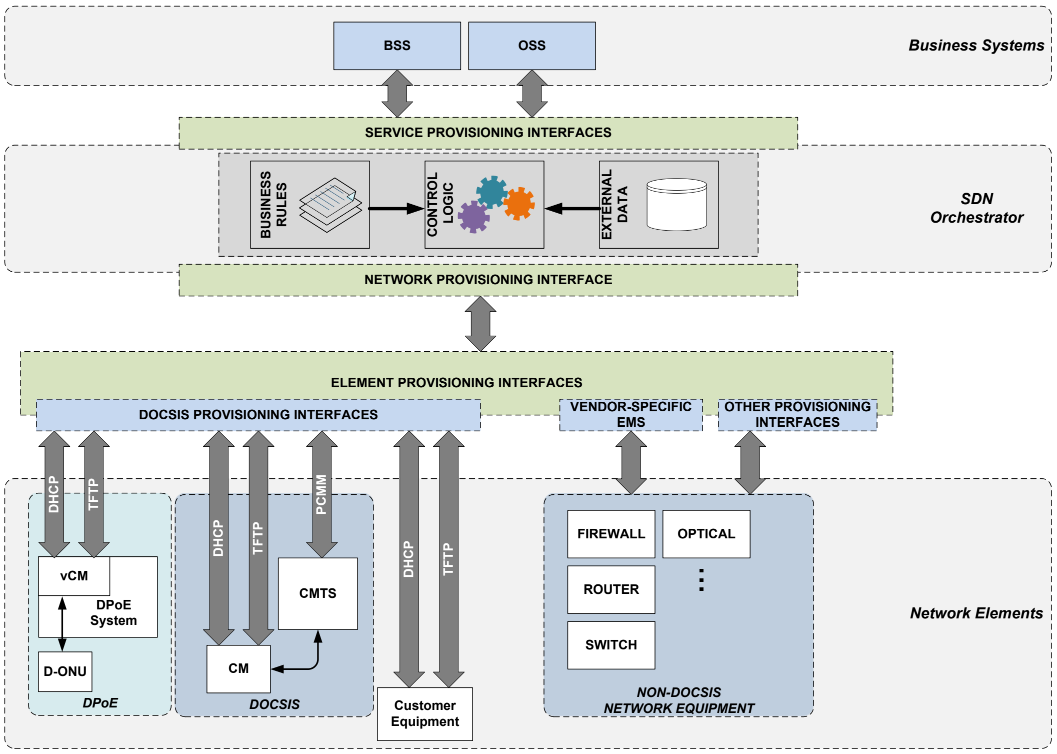

Figure 1 - SDN Architecture with DOCIS-based Provisioning

Automation is something that should enable flexibility and speed when it comes to defining services in the network, whether configuring and deploying existing services, chaining multiple services together to make a new integrated service, or rapidly building new and innovative services, as well as troubleshooting them when they break.

The remainder of this paper explores this aspect of SDN and how the combination of SDN and DOCSIS-based provisioning can help realize these goals.

An SDN Architecture using DOCSIS and DPoE

A provisioning architecture that incorporates SDN with DOCSIS and DPoE would position a cable operator to leverage the advantages of SDN without requiring a forklift upgrade of its existing access network equipment and front-office provisioning tools. Such an architecture can easily be envisioned and one is depicted in Figure 1.

It is important that the key requirements of such an architecture are understood.

- The SDN+DOCSIS architecture must be capable of provisioning the network services that are offered by the cable operator today. The network services typically being offered today are high-speed Internet access and Layer-2 VPN (ELINE, ELAN, ETREE) services.
- The architecture must be easily extended to support new network services (for example, L3VPN) in the future.
- The architecture must provide a rich business-facing API that is capable of describing the offered services without intimate knowledge of the underlying network. In fact, the API should not need to know the particular network technology that is in use.
- The architecture must provide the ability to chain network services to higher-layer services such as hosted

firewalls, application load balancing, cloud-computing resources, etc.

The Business Systems layer depicted in the architecture of Figure 1 is that set of applications and systems that are responsible for customer relationship management, billing, reporting, and workflow management. For most cable operators this will be their BSS (for example, Icoms and CSG). Given the weaknesses of traditional cable BSS to support Commercial Services some other system may interact with the architecture at this level.

The Service Provisioning Interface (SPI) is the API that business systems use to interact with the SDN Orchestrator. The Service Provisioning interface exposes a programmatic method by which the business systems can provide a "business-style" definition of the service(s) to be provisioned along with directives describing the action to be taken (move, add, change, delete, etc.) with the definition being provided to the SDN Orchestrator.

The SDN Orchestrator is responsible for translating the received service provisioning requests into a working network configuration. The SDN Orchestrator is expected to supplement the SPI input with business rules and data collected from the network and other sources.

Using all these inputs the SDN Orchestrator computes the most optimal network configuration that will meet the service requirements and pushes that configuration through the Network Provisioning Interface.

The Network Provisioning Interface (NPI) is responsible for the presentation of service-enabling parameters to the Element Provisioning Interface (EPI) that speaks directly to the network elements and their respective management systems.

The Element Provisioning Interface (EPI) implements the protocols that the individual network elements use to configure themselves for communication on the network and, if necessary, download a configuration from an element-provisioning server. The EPI will contain modules that are customized to the particular network element and manufacturer.

Ideally there would be only one module in the EPI. This would reduce the integration work required in the SDN systems over the long-term, but currently there is no single standard or protocol that can achieve complete coverage. As a result, most models are built with the expectation that multiple modules will be necessary, with the EPI serving as a method to abstract the element-specific configuration method(s) from the upper layers. Candidates for the EPI include currently developing standards like OpenDaylight as well as common off-the-shelf modular provisioning systems that leverage partnerships with network equipment vendors to build and maintain vendor-specific modules, or that use standard extensible configuration protocols such as NETCONF [RFC 6241] and YANG [RFC 6020].

The authors claim here and provide evidence in later sections that DOCSIS-style provisioning implemented in the EPI can carry a significant portion of the service-level provisioning functionality in the cable operator's network, and the presence of an EPI as an abstraction layer can allow support for other non-DOCSIS provisioning methods in concert to provide a complete end-to-end service provisioning solution.

Requirements of a Service Provisioning API

It is important that the API provided by the SPI create an abstraction between the business systems and the provisioning systems such that network-specific knowledge is not required in the business systems. When provisioning network transport services

(Internet access, Layer-2 VPN, etc.), when viewed from a business-layer perspective, it is sufficient to provide only the details required to describe (not completely specify) an attachment circuit and the backbone transport associated with each subscribed service in order to completely specify the end-to-end service.

The authors propose here that a network-layer service can be fully specified by the business systems by providing the following elements:

1. Unique service identifier that is common among the end points attached to a service
2. Specification of the service type (for example: IP, L2VPN, L3VPN)
3. Specification of the role that the end point plays in the service (for example: node, root, leaf)
4. Specifications of supplements to be instantiated on the service (for example: [Y1731], or [1588v2])
5. Unique Identifier of the Access Equipment (described below)
6. Port Identifier on which the service is expected to appear
7. Description of user traffic to accept and/or not accept into the service
8. Bandwidth profile for both upstream and downstream flows

Note that these attributes do not contain any data or imply any *a priori* knowledge of the network infrastructure *other than* an assumption that (1) the business systems have chosen the most appropriate access technology to be used (which could be done in the SDN Orchestrator) and (2) the network can be provisioned purely through communication with the end points (as in [DOCSIS]) or an entity acting on the end point's behalf (as in [DPOE]).

This model is not limited to [DOCSIS] and [DPOE] access network technologies. [EXPO2012] proposed a provisioning model for point-to-point Ethernet based on [DOCSIS] and an implementation of that proposal has been demonstrated [OLIVERDEMO] with common off-the-shelf routers and switches.

Of course, the list of elements above is not sufficient to describe higher-layer services (hosted firewalls, cloud computing resources, etc.). Any Service Provisioning API would need to provide a robust and extensible language to describe those services as well as methods to chain network transport services to those higher-layer services. This present paper will not attempt to address the requirements for those types of services because the authors recognize the insufficiency of DOCSIS-style provisioning to fully describe those services in an effective manner.

Further requirements of the API should specify the actions that can be performed through the API. For example, the API needs to allow the business systems to specify actions of Move, Add, Change or Delete (MACD). The API also needs to be queryable – in other words the business systems must be able to query the status of the requested service instance and the general network through the API.

The API must also allow flexibility in the actual implementation of the requested services. For example, the network operator may choose a network architecture that permits multiple services to be implemented on the same physical port. In this use case, the API must not make assumptions that a single physical port can have only one service. In fact, the SDN Orchestrator would use the business rules as input to determine whether multiple services are allowed on a single port or not.

This is a high-level and incomplete list of requirements for the Service Provisioning

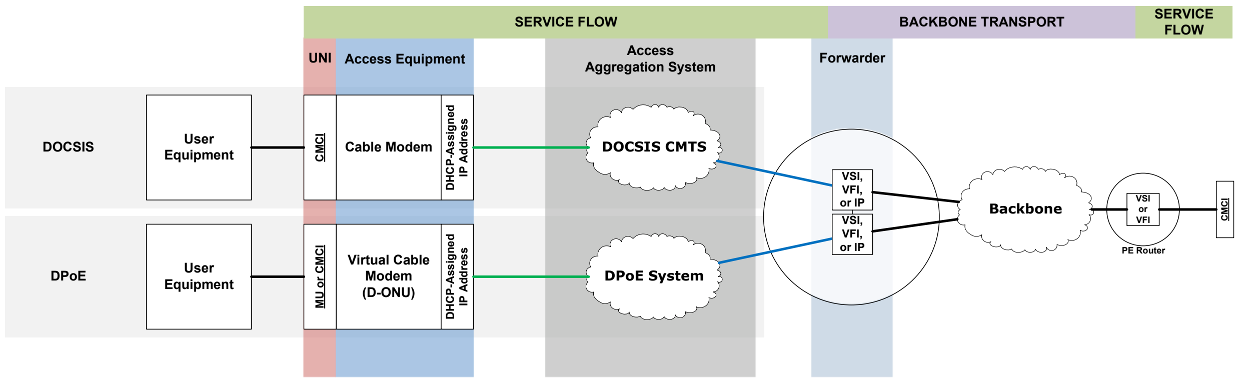

Figure 2 - Network Model for DOCSIS/DPoE Provisioning

API. They do, though, provide a good starting point for discussion of a more detailed set of requirements.

DOCSIS-STYLE PROVISIONING

The DOCSIS specification defines a high-speed data access network operating on a coaxial cable plant. Embedded in [D3-MULPI] is an element and network-provisioning framework. The following sections are a high-level tutorial of the DOCSIS provisioning model and are designed to support the claim that DOCSIS-style provisioning is capable of provisioning a rich and extensible set of services and fits well with the SDN model.

Of particular interest in this study is [D3-MULPI], [L2VPN], and [DPOE]. Specified within these documents is a network model, and protocol for initial provisioning of a cable modem (virtual cable modem in [DPOE]) and the service-description language to be used when configuring access services on a DOCSIS-style network.

Network Model

Rather than choosing to define a common network reference model to be applied to all the specifications, Cablelabs has chosen to weave a network model into each specification, [DOCSIS], [DPOE] and [L2VPN]. Luckily each model is very similar to the others and all are based largely on the [DOCSIS] model. Here we summarize the major elements in each specification.

The network model, shown in Figure 2, consists of:

- *User Equipment* – For example, a PC or router.
- *Network Access Equipment* – Equipment that is specific to the network type and associated with a subset of subscribers on the access network. A Cable Modem in a DOCSIS network, or a D-ONU in a DPoE network.
- *Access Aggregation System* – Equipment specific to the network type and aggregating users from one or more Network Access Equipment and typically located in an MSO's headend or hub. This would be a CMTS or DPoE System.
- *Forwarder* – A logical entity that is typically located in the Access Aggregation System. The Forwarder is responsible for moving customer data from the access network ports to a Network Side Interface (NSI). Forwarders could take the form of an IPv4 or IPv6 interface, an IEEE 802.1 bridge, an MPLS virtual interface, etc.

Within this model, user data is accepted on the CMCI (or MU in DPoE) and must be transported across the access network to the access aggregation system. Inside the access aggregation system, the user's traffic must be matched to a Forwarder instance.

The Forwarder then acts on the user's traffic according to the forwarder's configuration. An IPv4 forwarder will inspect the user's packets and route them according to the rules of IPv4 and the Forwarder's configuration. Similarly, an L2VPN forwarder will encapsulate the user's frames and bridge them according to the configuration assigned to that user and the forwarder.

A *Service Flow* is the construct that directs a user's data (or subset thereof) from the CMCI to the proper Forwarder instance. User data is classified as it enters the CMCI. Based on the classification, a service flow is matched and the frames are encapsulated and forwarded according to that service flow's configuration.

The preceding text describes traffic flowing upstream, meaning from the user toward the Network Services Interface (NSI). Treatment of traffic flowing from the NSI toward the user (downstream) follows a similar path. Traffic entering the NSI must be received by a Forwarder, classified into a service flow, forwarded through the service flow to the CMCI and delivered to the user. The remainder of this paper will describe traffic flow in the upstream reference, but the discussion applies equally to downstream flows.

DOCSIS Provisioning Framework

Any provisioning framework must provide two essential functions. The first is a language with which to describe the configuration to be applied to a network element. The second element is a method to deliver the configuration to the element being provisioned.

The original architects of [DOCSIS] chose a provisioning model in which the cable modem is the primary consumer of the service configuration (referred to in this document as *end-point provisioning*). This choice and the list of available protocols at the time sent the designers down a path of using DHCP and TFTP as the delivery mechanism. DHCP is required because the cable modem must have an IP address to be manageable on the network and TFTP because it is a simple protocol to implement for file transfers.

The configuration language chosen in [DOCSIS] is a Type-Length-Value (TLV) system. The Type indicates a parameter that is to be specified (for example, Bandwidth). The Value indicates the value being assigned to that instance of the Type. Length simply indicates how long the data is for the Type being specified – aiding the receiver in decoding the configuration file.

(In reality, the DOCSIS architects chose the system of TLV configuration files delivered via DHCP/TFTP because early manufacturers of data-over-cable equipment had developed and cable operators had already deployed systems based on these methods.)

The same system is, of course, adopted by [DPOE] for EPON networks.

The provisioning model used by [DOCSIS] and [DPOE] makes several key assumptions about the network and the elements that compose it.

First, [DOCSIS] is only intended to configure a network element at the service-specific level. In other words, there is no method in the [DOCSIS] model to provision the operational aspects of a CMTS or, in [DPOE], a DPoE System. For example, the

DOCSIS configuration file is not able to carry information to create or configure a cable interface on a CMTS. This function is best served by other methods like Command-Line Interface or systems based on NETCONF [RFC 6241] and YANG [RFC 6020].

Second, the end-point based model assumes that the cable modem (CM) or virtual cable modem (vCM) will be the primary consumer of the configuration file. In other words, the CM or vCM is the element that initiates the configuration file download and will convey the required information to other network elements (this is not true in the strictest sense in [DOCSIS] – the CMTS may actually acquire the configuration file contents by capturing the TFTP packets as they are sent to the cable modem).

Finally, the end-point based model assumes that any end-point may cause a forwarder to be instantiated by providing the essential parameters for the forwarder via the end-point's configuration file.

The model described here accomplishes initial provisioning of a user's service. If a change needs to be made to the service instance, it is necessary to reboot the access equipment. A weakness that may be perceived in the [DOCSIS] provisioning model is the lack of a dynamic method of updating a service.

PacketCable Multimedia (PCMM) is the first attempt to address this in the CableLabs specifications. The use of PCMM has not, yet, been defined in [DPOE]. PCMM only has a data model to describe changes in bandwidth or to setup and destroy service flows. PCMM does not currently have a data model to describe Forwarder instances. The latter point is significant in the Business Services space.

It is typical for an IP forwarder instance for Internet access to be pre-configured on the CMTS. Therefore PCMM simply needs to be able to configure service flows that will attach to that forwarder. However, in the Business Services space, it may be necessary to create a new forwarder, modify an existing forwarder, or attach a service flow to a specific forwarder. PCMM cannot perform this function today.

[DPOE] has defined a dynamic update method based on the initial provisioning mechanism. In this method, an external system sends an SNMP SET message indicating to the vCM that it must re-download its configuration file. Upon receipt of the SNMP SET, the vCM executes its initial provisioning sequence (DHCP and TFTP) *but does not reset*. The D-ONU and DPoE System continue forwarding traffic according to the previous configuration. Upon receipt and validation of the new configuration, the D-ONU and DPoE System are reconfigured and begin forwarding according to the new configuration.

DOCSIS Toolbox

The DOCSIS provisioning language, including extensions in [DPOE] and [L2VPN], currently is capable of describing a user's attachment to natively routed IP services and layer-2 VPN services. It is easy to conceive that the TLV dictionary could be extended to represent layer-3 VPN or other service types.

The following sections provide a *sampling* of the key elements contained in the current DOCSIS TLV dictionary. This section is not intended to provide an exhaustive list and description of the TLV dictionary. Interested readers should review [DOCSIS], [DPOE] and [L2VPN].

Equipment Identification

During provisioning it is necessary to identify the consumer of the provisioning data and the equipment to be provisioned. In

[DOCSIS] the consumer of the provisioning data and the equipment to be provisioned are the same – they are both the cable modem. In [DPOE] the vCM is the consumer and the D-ONU is the equipment, but they appear as the same entity to the DOCSIS-based provisioning system.

MAC Address

[DOCSIS] uses the 48-bit IEEE 802.3 MAC Address of the cable modem's RF interface for equipment identification.

[DPOE] uses the 48-bit IEEE 802.3 MAC Address of the D-ONU as the equipment identifier.

The MAC address is used during the DHCP process (in the BOOTP chaddr field) to identify the consumer of the configuration file (the cable modem in [DOCSIS] and the vCM in [DPOE]). Because the provisioning consumer and the provisioned equipment are the same, it is not necessary, though it may be desirable, to use the MAC address in the cable modem configuration file.

CMIM

In some use cases it is also necessary to identify the specific port to which a user is being attached. For example, one port on a multi-port D-ONU may provide Internet access to subscriber-A and another port may provide Layer-2 VPN service to another subscriber. Within the configuration file, the CMIM TLV is used to associate the port to the service flow.

<u>*Classification*</u>

Classification refers to the process of inspecting incoming packets or frames and associating them with an action based on their content. There are two possible classifier actions in [DOCSIS] and [DPOE] – packets can be dropped or they can be forwarded into a service flow.

Classification occurs independently in the upstream and downstream direction; therefore the DOCSIS provisioning dictionary defines two different classifier types. The classifier TLVs are containers for a rich set of classification criteria.

Classification Criteria

Classification criteria are available for all IPv4 and IPv6 headers, TCP and UDP headers, Ethernet MAC headers and MPLS headers. The complete list of available classification criteria are cataloged in Annex C of [D3-MULPI]. Some examples of particular interest to providing business services are:

CMIM – Specifies the access equipment's physical port on which user traffic is expected to enter the network. Used in upstream classification.

IEEE 802.1ad C-VID – Specifies the customer-supplied VLAN ID (in Provider Bridging format) to use for classification. Typically (but not necessarily) used in upstream classifiers.

IEEE 802.1ad C-PCP – Specifies the customer-supplied CoS markings (in Provider Bridging format) to use for classification. Typically (but not necessarily) used in upstream classifiers.

IEEE 802.1Q VLAN-ID – Specifies the customer-supplied VLAN ID (in the pre-802.1ad format) to use for classification. Typically (but not necessarily) used in upstream classifiers.

IEEE 802.1P User Priority – Specifies the customer-supplied CoS markings (in the pre-802.1ad format) to use for classification. Typically (but not necessarily) used in upstream classifiers.

IPv4 ToS – Specifies the IPv4 ToS markings to match in classification. Typically (but not necessarily) used in upstream classifiers.

IPv6 Flow Label – Specifies the IPv46 flow label to match in classification. Typically (but not necessarily) used in upstream classifiers.

IPv6 Traffic Class – Specifies the IPv6 traffic class markings to match in classification. Typically (but not necessarily) used in upstream classifiers.

MPLS Label – Specifies the MPLS label to match in classification. Typically (but not necessarily) used in downstream classification.

MPLS Traffic Class – Specifies the MPLS Traffic Class (also known as the EXP bits) to match in classification. Typically (but not necessarily) used in downstream classification.

<u>*Service Flow Description*</u>

A Service Flows is the construct that describes the treatment of a specific classified flow of packets as it is transmitted across the access network. The service flow also maintains the association of the packets to the Forwarder instance in the access aggregation system. Treatment includes the application of bandwidth profiles and could include traffic coloring and re-marking.

Bandwidth Profile

The encodings of most interest in the service flow description are those that describe the *Bandwidth Profile*. The bandwidth profile can take one of two forms. The DOCSIS QoS profile is the most commonly used and can contain settings for Traffic Priority, Maximum Sustained Traffic Rate, Maximum Sustained Traffic Rate, Maximum Traffic Burst, Minimum Reserved Traffic Rate and others.

The second form for a bandwidth profile is the *Metro-Ethernet Service Profile* (MESP). The MESP is based on [MEF10] and contains settings for Committed Information Rate (CIR), Committed Burst Size (CBS), Excess Information Rate (EIR), and Excess Burst Size (EBS) and the color mode.

<u>*Forwarder Description*</u>

Currently there are three Forwarder types in [DOCSIS], [DPOE], and [L2VPN]: IPv4 and IPv6 forwarders, MPLS forwarders, and layer-2 forwarders (typically 802.1Q-based bridges).

IPv4 and IPv6 Forwarders

In the DOCSIS TLV dictionary there are no TLVs to describe an IPv4 or IPv6 forwarder. Rather, IPv4 and IPv6 forwarders are assumed to be pre-configured and are the default forwarder on the system. That is to say, a service flow will be associated with an IPv4 or IPv6 forwarder unless there is an alternate association specified in the downloaded configuration file. It is possible that multiple IPv4 and IPv6 forwarders exist, therefore it would be necessary to specify a Service Class Name or Attribute Mask TLV to steer the service flow to the intended forwarder instance.

Layer-2 Forwarders

Layer-2 Forwarders are intended to simply pass Ethernet frames received at the CMCI to an external entity using MAC-layer bridging rules. To accomplish this, the forwarder requires a description that simply contains an NSI Encapsulation to specify the layer-2 attributes. Some key TLVs available to configure layer-2 forwarders are:

IEEE 802.1Q Encapsulation – This encapsulation setting defines a layer-2 forwarder with 802.1Q tagging.

IEEE 802.1ad Encapsulation – This encapsulation setting defines a layer-2 forwarder with Provider Bridge tagging.

IEEE 802.1ah Encapsulation – This encapsulation setting defines a layer-2 forwarder with Provider Backbone Bridge tagging.

MPLS Forwarders

MPLS Forwarders are intended to encapsulate into MPLS packets arbitrary frames or packets received through an associated service flow. The forwarders can be configured to create explicitly defined pseudowires or they can be configured with parameters to enable BGP auto-discovery, BGP signaling or LDP signaling.

To create an MPLS Forwarder the NSI Encapsulation must contain the proper combination of the following TLVs:

MPLS Pseudowire ID – Defines the primary pseudowire identifier.

MPLS Peer IP address – Specifies the IP address of the MPLS PW peer.

Pseudowire Type – Specifies whether the forwarder is a member of a VPLS or (for point-to-point pseudowires), whether the PW is in Ethernet tagged mode or Raw mode.

L2TPv3 Forwarders

Similar to MPLS forwarders, L2TPv3 Forwarders encapsulate arbitrary frames or packets into an L2TPv3 tunnel.

<u>*End-to-End Service Setup*</u>

When dynamic pseudowire signaling (LDP or BGP) and auto-discovery (BGP) are used, it is necessary to provide the forwarder the necessary information to perform these functions.

Attachment Group ID – Specifies the Attachment Group ID (AGI) used in dynamic MPLS and L2TPv3 signaling of point-to-point pseudowires.

Source/Target Attachment Group ID – Specifies the Source and/or Target AGI for dynamic signaling of point-to-point pseudowires.

BGP VPNID – Uniquely identifies the service (VPN) to the auto-discovery protocols. This value is unique to the service, but common among all attachments to the service.

Route Distinguisher – Specifies one or more route distinguishers (RD) to be associated with the service and advertised by this particular attachment (via BGP) to the service.

Route Target (import/export) – Specifies a list of route targets (RT) to be imported from and/or exported to the BGP database.

<u>*Supplemental Service Configurations*</u>

The TLV dictionary categories discussed above are concerned with establishing connectivity between the user's equipment and the forwarder. Subscribers often require more than simple connectivity, though. The DOCSIS TLV dictionary accommodates supplemental configurations, also.

Two features of recent interest to operator's Commercial Services business are timing and Service OAM. The DOCSIS TLV dictionary contains the constructs necessary to configure distribution of the [1588v2] Precision Time Protocol across the access network. The dictionary also contains the constructs to define ITU Y.1731 Maintenance Entity Groups (MEG) [Y1731].

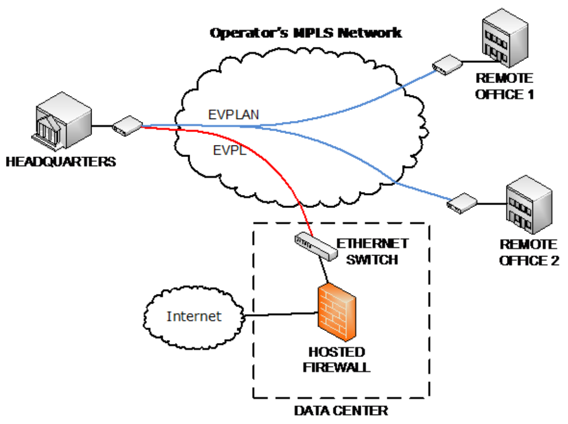

Figure 3 - Example Customer Network Configuration

DOCSIS AND SDN IN CONCERT

The early discussion in this paper described the overall architecture and the key elements of that architecture. Later, the discussion moved to requirements of the service provisioning API and the capabilities of DOCSIS-style provisioning.

What about the practical reality of this combination when asked to provision real services being offered today by cable operators? Given a non-trivial use case (in other words, NOT high-speed Internet access), how would an SDN architecture using DOCSIS-style provisioning function?

Consider a typical Business Services customer that wishes to connect several sites using Ethernet over a Layer-2 VPN and attach an Internet access service to the same VPN. The proposed topology is shown in Figure 3.

In this network, the cable operator would need to provision four (4) end points and a firewall instance. The four end points are

1. D-ONU (assume a DPoE connection) at the Headquarters
2. Cable modem at Remote Office 1
3. D-ONU at Remote Office 2
4. Ethernet Switch Port in the operator's data center

The service-specific provisioning process begins with the business systems requesting service for the four end points and the hosted firewall. Following the SDN provisioning model described earlier the business systems would describe each site to the Service Provisioning API as follows:

EVPLAN at Headquarters

ATTRIBUTE	VALUE
SERVICE ID	CUSTOMER_1_SERVICE_1
SERVICE TYPE	L2VPN
SERVICE ROLE	ROOT
ACCESS EQ'T ID	<HQ D-ONU MAC>
ACCESS PORT ID	Port 1
TRAFFIC SPEC	User-Supplied C-TAG = 99
UPSTREAM BW	50Mbps
DOWNSTREAM BW	25Mbps

EVPLAN at Remote Site 1

ATTRIBUTE	VALUE
SERVICE ID	CUSTOMER_1_SERVICE_1
SERVICE TYPE	L2VPN
SERVICE ROLE	ROOT
ACCESS EQ'T ID	<SITE 1 CM MAC>
ACCESS PORT ID	Port 1
TRAFFIC SPEC	User-Supplied C-TAG = 99
UPSTREAM BW	5Mbps
DOWNSTREAM BW	25Mbps

EVPLAN at Remote Site 2

ATTRIBUTE	VALUE
SERVICE ID	CUSTOMER_1_SERVICE_1
SERVICE TYPE	L2VPN
SERVICE ROLE	ROOT
ACCESS EQ’T ID	<SITE 2 D-ONU MAC>
ACCESS PORT ID	Port 1
TRAFFIC SPEC	User-Supplied C-TAG = 99
UPSTREAM BW	5Mbps
DOWNSTREAM BW	25Mbps

EVPL at Headquarters

ATTRIBUTE	VALUE
SERVICE ID	CUSTOMER_1_SERVICE_2
SERVICE TYPE	L2VPN
SERVICE ROLE	NODE
ACCESS EQ’T ID	<HQ D-ONU MAC >
ACCESS PORT ID	Port 1
TRAFFIC SPEC	User-Supplied C-TAG = 35
UPSTREAM BW	10 Mbps
DOWNSTREAM BW	100 Mbps

EVPL at Data Center

ATTRIBUTE	VALUE
SERVICE ID	CUSTOMER_1_SERVICE_2
SERVICE TYPE	L2VPN
SERVICE ROLE	NODE
ACCESS EQ’T ID	<DC SWITCH MAC >
ACCESS PORT ID	Port 42
TRAFFIC SPEC	User-Supplied C-TAG = 35
UPSTREAM BW	100 Mbps
DOWNSTREAM BW	10 Mbps

The Service Provisioning API would need to expose a method to request the hosted firewall. This paper does not attempt to define this method or the data model for it because DOCSIS-style provisioning is clearly inappropriate for upper-layer services.

Next in the provisioning process the SDN Orchestrator, would process the service provisioning requests and any available data about the network through the business rules. Of course, by nature, the SDN Orchestrator would know the configuration and capabilities of the network and incorporate that knowledge in the processing. The output of this processing would be a set of decisions about how to configure the network to support the requested services.

The SDN Orchestrator would then use this set of decisions to push an abstract configuration set through the Network Provisioning Interface (NPI). The NPI would then translate that abstract configuration into a specific set of parameters to be fed into the DOCSIS Provisioning Interface.

The output of the NPI would be a set of DOCSIS configuration files with one crafted to match each site and the overall service configuration. We assume here that the

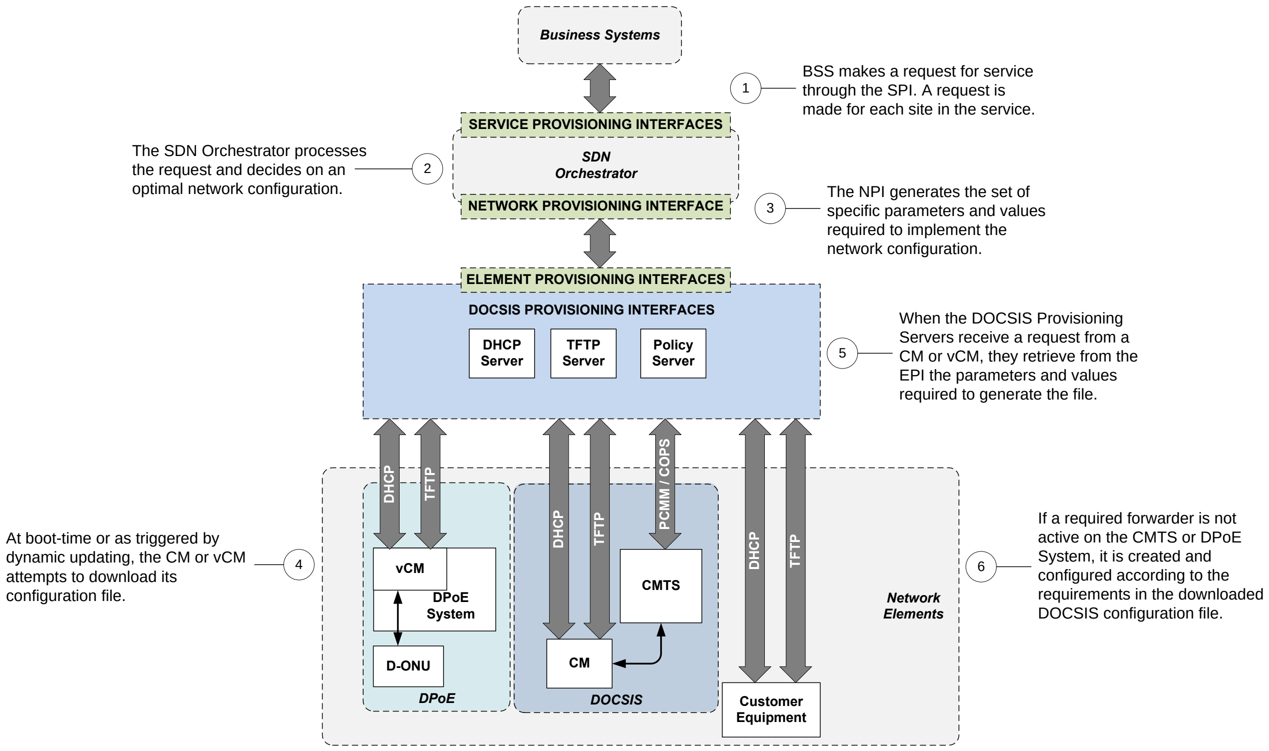
Business Systems
1
BSS makes a request for service through the SPI. A request is made for each site in the service.
SERVICE PROVISIONING INTERFACES
SDN Orchestrator
2
The SDN Orchestrator processes the request and decides on an optimal network configuration.
NETWORK PROVISIONING INTERFACE
3
The NPI generates the set of specific parameters and values required to implement the network configuration.
ELEMENT PROVISIONING INTERFACES
DOCSIS PROVISIONING INTERFACES
DHCP Server
TFTP Server
Policy Server
5
When the DOCSIS Provisioning Servers receive a request from a CM or vCM, they retrieve from the EPI the parameters and values required to generate the file.
DHCP
TFTP
PCMM / COPS
4
At boot-time or as triggered by dynamic updating, the CM or vCM attempts to download its configuration file.
vCM
DPoE System
D-ONU
DPoE
CMTS
CM
DOCSIS
Customer Equipment
Network Elements
6
If a required forwarder is not active on the CMTS or DPoE System, it is created and configured according to the requirements in the downloaded DOCSIS configuration file.

Figure 4 - Provisioning Flow through the SDN+DOCSIS Architecture

operator's network is configured to support VPLS and VPWS using explicitly defined peers. Using this assumption and the input provided through the Service Provisioning API, a set of DOCSIS configuration files can be constructed.

At boot-time the CM or vCM executes the procedures defined in [DOCSIS] or [DPOE], respectively. These procedures include DHCP to obtain an IP address and TFTP to download the configuration file. When the DHCP server or the TFTP server receive a request from the CM or vCM, they will retrieve from the Element Provisioning Interface (or the EPI will provide to them prior to the CM's attempt) the parameters needed to construct a response (DHCP) or the configuration file (TFTP) that will accomplish the intended network configuration.

A dynamic update could be signaled, whether via PCMM or the dynamic file update method defined in [DPOE]. In this case, the policy server (if PCMM) or the DHCP and TFTP servers (if dynamic file update) will retrieve the updated parameters from the EPI and respond back to the CM or vCM.

Once the configuration file is downloaded (or the dynamic update is processed), the CM registers the configuration with the CMTS. The CMTS (or DPoE System) examines the configuration to determine which forwarder to which the defined service flow(s) should be attached. If the forwarder does not exist, then the CMTS will create the forwarder according to the parameters contained in the configuration file.

An example configuration file for the D-ONU located at the headquarters is given below.

```
#BEGIN
Network Access = 1
# Constructs for EVPLAN
US Classifier
 SF Ref = 1
 802.1ad Classifier
   CVID = 99
```

```
DS Classifier
 SF Ref = 2
 802.1ad Classifier
   SVID = 1100
 L2VPN Encoding
   Vendor ID = FFFFFF
   VPNID = "CUSTOMER_1_SERVICE_1"

US Service Flow
 Ref = 1
 QOS ParamSetType = 7
 Min Reserved Rate = 50000000
 Max Sustained Rate = 50000000
 L2VPN Encoding
   Vendor ID = FFFFFF
   VPNID = "CUSTOMER_1_SERVICE_1"

DS Service Flow
 Ref = 2
 QOS ParamSetType = 7
 Min Reserved Rate = 25000000
 Max Sustained Rate = 25000000

L2VPN Encoding
  Vendor ID = FFFFFF
  VPNID = "CUSTOMER_1_SERVICE_1"
  L2VPN Mode = Encapsulation
  NSI Encapsulation Type = MPLS
Pseudowire
  MPLS Pseudowire Type = VPLS
  MPLS Pseudowire ID = 650001100
  MPLS Peer IP Address =
IPv4:192.168.100.1
  MPLS Peer IP Address =
IPv4:192.168.101.1
# end EVPLAN
# Constructs for EVPL
US Classifier
 SF Ref = 3
 802.1ad Classifier
   CVID = 35

DS Classifier
 SF Ref = 4
 802.1ad Classifier
   SVID = 1101
 L2VPN Encoding
   Vendor ID = FFFFFF
   VPNID = "CUSTOMER_1_SERVICE_2"

US Service Flow
 Ref = 3
 QOS ParamSetType = 7
 Min Reserved Rate = 10000000
 Max Sustained Rate = 10000000
 L2VPN Encoding
   Vendor ID = FFFFFF
   VPNID = "CUSTOMER_1_SERVICE_2"

DS Service Flow
 Ref = 4
 QOS ParamSetType = 7
 Min Reserved Rate = 100000000
 Max Sustained Rate = 100000000

L2VPN Encoding
 Vendor ID = FFFFFF
 VPNID = "CUSTOMER_1_SERVICE_2"
 L2VPN Mode = Encapsulation
 NSI Encapsulation Type = MPLS
Pseudowire
 MPLS Pseudowire Type = Ethernet Raw
Mode
 MPLS Pseudowire ID = 650001101
 MPLS Peer IP Address =
IPv4:192.168.102.1
# end EVPL
# EOF
```

The flow through the SDN architecture is depicted in Figure 4.

GAPS, CHALLENGES AND THE FUTURE

DOCSIS/DPOE can be used to provision services, such as L2VPN or Internet Access, that are commonly provided on an MSO’s network. Most services can be provisioned via DOCSIS in such a way that no configuration beyond what is gleaned from the DOCSIS configuration file is necessary in the upstream network devices. In this model, the endpoint needs to know very little about the network between it and its remote-side destination(s). Likewise, the network needs to maintain very little state about the underlying service and its topology.

However, there are significant gaps in the support for provisioning more advanced services like L3VPN.

To support L3VPN, additional TLVs would need to be added to the DOCSIS provisioning dictionary to define

- L3VPN topologies (for example, Hub and spoke, partial mesh, NNI/Extranet)
- PE-CE routing protocol configuration
- Route policies needed to ensure the correct routes are announced, filtered, and tagged

Additional gaps exist when considering how to define more complex and chained services, such as providing inline firewall services, or providing access to cloud services from within a VPN. Higher layer services like these are examples of services where the service definition may extend beyond attachment circuits. In L3VPN, elements in the network provide a Layer 3 routed topology, participate in the routing protocol, and require more state exchange between the end point and the network, so in some ways the DOCSIS provisioning model may be inadequate.

One might reasonably ask why the existing framework around DOCSIS/DPOE could not simply be extended to include additional service provisioning since it's intended to be extensible. There is a point at which adding into the DOCSIS provisioning dictionary the TLVs necessary to represent more complex service definitions reaches diminishing returns, especially when considering the need to provision non-network services.

It may be conceivable that the DOCSIS-style configuration file and dictionary could represent virtually any service one might wish to describe. However, when multiple service elements are involved, especially if one or more are separate from the access termination, the constructs of a DOCSIS configuration file are likely to be less rich than modern configuration languages like NETCONF/YANG ([RFC 6020], [RFC 6241]), OpenFlow, or vendor-specific element management APIs.

Further, the DOCSIS method of delivering configuration files is not supported by many systems. While it could be adapted, one is forced to consider whether the effort is deserved over other available methods. In fact, the proposed architecture assumes that there are multiple methods at the Element Provisioning layer and that the SDN Orchestrator would generate the "glue" to chain individual element configurations into an end-to-end service.

The lack of support for DOCSIS-style provisioning in elements beyond cable modems, CMTSes, DPoE Systems and vCMs highlights another gap. Today [DOCSIS] is a monolithic specification. Included in that specification is the MAC and PHY requirements necessary to transmit data over an HFC network and service definitions and provisioning for all of the potential services that use that transmission technology.

It is not reasonable to expect a network equipment manufacturer to understand the entirety of [DOCSIS] if they do not support DOCSIS MAC/PHY interfaces. Yet it is difficult work to identify the specific elements of [DOCSIS] that are required to support DOCSIS-based provisioning without supporting the data transmission protocol.

[DPOE] is a step in the direction of separating DOCSIS provisioning from DOCSIS data transmission, but it accomplishes this not by extracting the provisioning elements from the specifications to make them standalone, but by emulating many of the MAC-layer elements of [DOCSIS] such that the protocol continues to work as-is. This approach works very well for most access-layer network technologies, but extending this to support a wider range of services would require that DOCSIS-style provisioning be separated into a specification of its own.

CONCLUSION

In the combined SDN+DOCSIS model presented here, the service definition and topology are abstracted from the physical access and the devices used to provide the service. This abstraction allows for maximum flexibility in building a provisioning system that is agnostic to the access technologies being used.

As MSOs consider the future of their networks and services, deploying an architecture such as the one proposed in this document allows them to build on existing capabilities while keeping flexibility to evolve the methods used to manage the network and service offerings. Manual provisioning of commercial services that cannot be provisioned via the existing DOCSIS infrastructure today is not a scalable and sustainable strategy.

DOCSIS-based provisioning systems manage hundreds of millions of devices around the world and process thousands of transactions per day. It is a tested and proven system. As cable operators look to expand into the $130 Billion Commercial Services market they will need to find ways to distinguish themselves over competitive operators.

DOCSIS-based provisioning is more than capable of provisioning the transport services offered by MSOs today. The potential exists to extend the DOCSIS TLV dictionary to support additional transport services. However, adding support for upper-layer services like cloud-computing into the DOCSIS provisioning framework requires careful evaluation and probably makes no sense.

By leveraging an SDN model with DOCSIS-based provisioning, the cable operator can reduce their time to bring new services to market, eliminate manual and error-prone configuration of network equipment, and increase the rate at which new customers can be added to the network. Especially in the context of complex Commercial Business offerings, the cable operator is able to reduce the service provisioning process to a transaction rather than a complex series of steps involving multiple systems and humans. These factors will allow the MSO to offer better-quality services at lower cost – making the cable operator a more desirable network provider.

[DPOE-MULPI] DOCSIS® Provisioning of EPON Specifications, DPoE™ MAC and Upper Layer Protocols Requirements, DPoE-SP-MULPIv1.0, Cable Television Laboratories, Inc.

[L2VPN] Data-Over-Cable Service Interface Specifications, Business Services over DOCSIS, Layer 2 Virtual Private Networks, CM-SP-L2VPN, Cable Television Laboratories, Inc.

[DPOE] Refers to the collective set of DPoE Specifications (all versions)

[D3-MULPI] Data-Over-Cable Service Interface Specifications DOCSIS 3.0, MAC and Upper Layer Protocols Interface Specification, CM-SP-MULPIv3.0, Cable Television Laboratories, Inc.

[DOCSIS] Refers to the collective set of DOCSIS and L2VPN Specifications (all versions)

[8021D] IEEE Std. 802.1D-2004, IEEE Standard for Local and Metropolitan Area Networks – Media Access Control (MAC) Bridges, June 2004.

[8021Q] IEEE Std. 802.1Q-2011, IEEE Standard for Local and Metropolitan Area Networks – Media Access Control (MAC) Bridges and Virtual Bridge Local Area Networks, August 2011.

[1588v2] IEEE Std. 1588™-2008, IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems.

[MEF10] MEF Technical Specification MEF 10.2, Ethernet Services Attributes Phase 2, Metro-Ethernet Forum, 27 Oct. 2009

[Y1731] ITU-T Recommendation Y.1731 (07/11), OAM functions and mechanisms for Ethernet based networks

[OLIVERDEMO] http://btreport.net/2013/10/oliver-demos-sdn-docsis-provisioning/

[EXPO2012] N. Nadiraju, K. Noll, "Transforming 'DOCSIS Provisioning of EPON' to 'DOCSIS Provisioning of Everything'", SCTE Cable-Tec EXPO Proceedings, October 2012.

[RFC 6241] IETF RFC 6241, Network Configuration Protocol (NETCONF), June 2011

[RFC 6020] IETF RFC 6020, YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF), October 2010