International Journal of Web Services Research

January-March 2015, Vol. 12, No. 1

Table of Contents

Special Issue on New Techniques of Services Computing

Guest Editorial Preface

The *International Journal of Web Services Research* is indexed or listed in the following: ABI/Inform; ACM Digital Library; Bacon's Media Directory; Burrelle's Media Directory; Cabell's Directories; Compendex (Elsevier Engineering Index); CSA Illumina; Current Contents®/ Engineering, Computing, & Technology; DBLP; DEST Register of Refereed Journals; Gale Directory of Publications & Broadcast Media; GetCited; Google Scholar; INSPEC; Journal Citation Reports/Science Edition; JournalTOCs; Library & Information Science Abstracts (LISA); MediaFinder; Norwegian Social Science Data Services (NSD); PubList.com; Science Citation Index Expanded (SciSearch®); SCOPUS; The Index of Information Systems Journals; The Standard Periodical Directory; Thomson Reuters; Ulrich's Periodicals Directory; Web of Science

Software-Defined Networking in Access Networks: Opportunities, Challenges, and Choices

Chen Tian, Huazhong University of Science and Technology, Wuhan, China

Jie Wu, Fudan University, Shanghai, China

Haibin Song, Huawei, Nanjing, China

ABSTRACT

Software-Defined Networking proposes to fundamentally change the current practice of network control. The two basic ideas are Centralized State Control and Uniform Device Abstraction, which support the Software-Defined promise. SDN has made significant progress. The opportunities of SDN in carrier access networks have been largely ignored by both industry and academia. In access networks, Quality-of-Service (QoS) oriented bandwidth management is more critical; the flexible QoS provisioning could be the most important opportunity for SDN. In this position paper, the authors show that the unique characteristics of access networks pose significant challenges to the two basic ideas. Contrary to the common agreement on "match-action" abstraction, the authors argue that the object-oriented abstraction might be a better choice for access networks to make a better software-defined implementation.

Keywords: *Access Network, NETCONF, Object-Oriented Abstraction, Openflow, Quality-of-Service, Software-Defined Networking, YANG*

INTRODUCTION

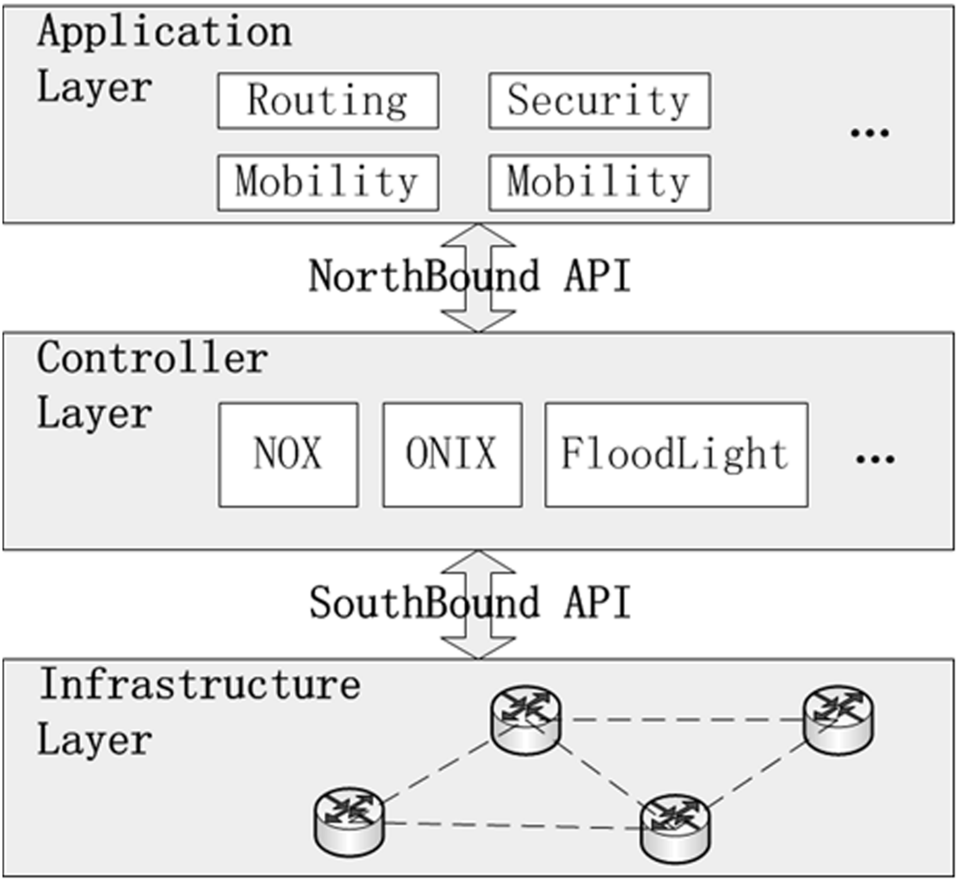
Software-Defined Networking (SDN) proposes to fundamentally change the current practice of network control (McKeown 2008; Shenker 2011). In recent years, there are significant research and implementation efforts for SDN from both industry and academia (Casado 2007; Sherwood 2009; Canini 2012; Monsanto 2013; Yu 2013; Jain 2013). The most prominent SDN achievement is the B4 Project (Jain 2013), where Google uses SDN to perform traffic engineering for its global inter-datacenter networks.

Shown in Figure 1 (Sezer&Scott-Hayward, 2013), the SDN architecture has 3 layers: Application layer, Control layer and Infrastructure layer. The philosophy of SDN is that basic state distribution primitives should be implemented only once rather than separately for every control task.

Leading by Open Networking Foundation (ONF), SDN promotes two basic ideas (Sezer&Scott-Hayward, 2013):

DOI: 10.4018/IJWSR.2015010101

Figure 1. SDN functional architecture



- **Centralized State Control:** For Control layer, a physically separated and logically centralized control platform handles state collection from all devices, make decisions, and distributes the control state to them.
- **Uniform Device Abstraction:** For Infrastructure layer, devices of the forwarding plane could be controlled by a uniform open interface, which also removes the danger of vendor lock-in.

These two ideas together support the *Software-Defined* promise in Application layer: a fully programmatic interface upon which developers could build network management applications on (Koponen&Casado, 2010).

SDN has made significant progress. ONF advocates OpenFlow as the standard southbound interface defined between the Control and Infrastructure layers. In Control layer, there are many controllers emerged such as NOX (Gude&Koponen, 2008), ONIX (Koponen 2010) and Maestro (Cai&Cox, 2010). The OpenDaylight project promises to unify the northbound API between the Control and Application layers (Gopal, 2013). An additional OpenFlow management and configuration protocol is also proposed to remotely configure

Table 1. Acronyms

BRAS	Broadband Remote Access Server.
BNG	Broadband Network Gateway.
DSLAM	Digital Subscriber Line Access Multiplexer.
RG	Residential Gateway.
OLT	Optical Line Terminal.
ONU	Optical Network Unit.

the control channel between the controllers and switches.

Besides its success in data-center and enterprise networks, people also realize the value of SDN in carrier networks (Elby, 2011), mostly in transport area (Mcdysan, 2013, pp.28-31). As a comparison, opportunities of SDN in access networks have been largely ignored by both industry and academia.

Access networks are different from datacenter and enterprise networks. Devices in those networks (*i.e.*, switches and routers) focus on forwarding, which is well suited for the common ''match-action'' SDN abstraction. While in access networks, Quality-of-Service (QoS) oriented bandwidth management among subscribers/services is more critical: rate limiting and hierarchical bandwidth scheduling are basic requirements to operational management.

Flexible QoS provisioning is the best opportunity for SDN in this area. Critical to increase ISP revenue, most value-added services require QoS guarantee (Matsumoto, 2012). From a broad investigation of operators and vendors, we found that the two major obstacles are (1) inconvenience in network configurations, and (2) vendor device diversity. SDN is exactly the right answer to flexible QoS provisioning.

However, the unique characteristics of access networks also pose challenges to the two basic ideas of SDN. The first is that ''not everything can be centrally controlled''; the second is that the simple ''match-action'' abstraction may not be appropriate for the southbound control interface in access scenarios.

This paper is, instead of a technical contribution, but a ''call to arms'' for the SDN community to face both the opportunities and challenges in access networks. We present related efforts, and argue that the object-oriented abstraction might be a better choice.

BACKGROUNDS

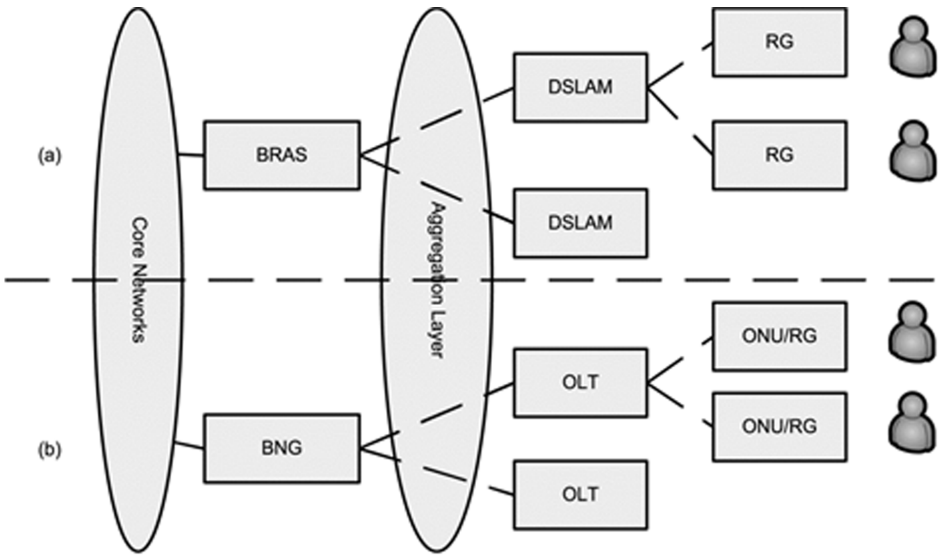
This part is a brief background of modern ISP access networks (Acronyms are listed in Table 1).

Typical Access Networks

There are two major access technologies: Digital Subscriber Line (DSL) and Passive Optical Network (PON). Note that there are other technologies such as direct ethernet connection and wireless 3G/LTE. We omit them here so we can focus on the major scenarios.

First let's look at the DSL scenario. As shown in Figure 2(a), each DSL subscriber has a RG in their home. Each RG connects to a DSLAM port via a telephone-line; the signal quality of the line is time-variable due to environment impacts; accordingly, DSLAM dynamically adjusts the transmission coding hence changes the uplink/downlink capacity. Each DSLAM connects to a BRAS via a metropolitan-scale aggregation layer network. The aggregation layer is usually ethernet-based and has a certain ratio of bandwidth oversubscription (Broadband Forum). BRAS is the network access point and connects to the ISP core networks. Given its abundant hardware capability, BRAS is where bandwidth limiting (*e.g.*, for an 4Mbps plan subscriber) and QoS scheduling enforced.

Figure 2. Access networks: (a) DSL (b) PON



Next is the PON scenario. As shown in Figure 2(b), each PON subscriber has an ONU/RG in their home. An OLT controls multiple ONUs in the same optical distribution network. The optical downlink is broadcast: each ONU receives all packets and filters based on destination ID. The uplink is shared: OLT controls the upload bandwidth allocation to different ONUs via Dynamic Bandwidth Allocation (DBA) (McGarry&Reisslein, 2012). Each ONU maintains multiple packet queues and reports the state to OLT; the OLT then grants transmission byte counts to each ONU separately. The role and connectivity of BNG is similar to that of BRAS in DSL scenario.

Need for Flexible QoS Provisioning

Given the prevalent flat-rate subscription mode in most ISPs, value-added services are critical to increase ISP revenue. One concrete example is Bandwidth-on-Demand (BoD): when a subscriber needs to watch a HD movie in peak hours, a paid BoD service could guarantee his bandwidth by reservation in the whole metro-network from BRAS to RG, regardless of the competition from other flows of other subscribers. Another example is Latency-on-Demand (LoD): extremely low latency needs to be guaranteed for a paid real-time online gamer, by given higher priority to his gaming packets over other traffic.

Nowadays, such value-added services are hard to be deployed. Our investigation of several typical metro-networks suggests that, QoS functionalities in existing devices are already sufficient for realizing those services.

There are two major obstacles. The first problem is the inconvenience in network configurations: the operator-centric configuration process is slow and error-prone even with the help of scripts. In one of our field test, a typical BoD provisioning requires around 100 lines of CLI in the BRAS alone. Sometimes, operators need to configure QoS capability in every single node alone a path one by one. Sometimes, new QoS rules even conflict with the existing configuration rules which require manual handling.

The second obstacle is vendor device diversity. Like that in other networks, current access devices as monolithic, closed, and mainframe-like. In our another trial, an ISP's LoD efforts finally failed because the same roles in different paths belong to different vendors, thus need quite different configuration processes.

We believe SDN is the right answer to flexible QoS provisioning: the centralized control idea could solve the first obstacle; the uniform abstraction could solve the second obstacle.

NOT EVERY STATE CAN BE CENTRALLY CONTROLLED

The unique characteristics of access networks also pose challenges to the two basic ideas. First of all, is that ''not every state can be centrally controlled''.

As mentioned above, the physically separated and logically centralized control platform is the leading idea of SDN. The situation is that there is some control state management intelligence which cannot be totally moved away to a physically separated entity. The reason is that access networks are geographically much larger than data-center networks; due to the latency between devices and controller, the control loop would be prolonged and the efficiency of the mechanism would degrade significantly. There are two kinds of such scenarios: interaction between nodes and instantaneous decision inside nodes.

Interaction between Nodes

A solid example is BRAS-DSLAM control loop. As mentioned above, the signal quality of a telephone-line is time-variable and DSLAM dynamically adjusts the downlink capacity. BRAS controls the download rate limiter to every DSL subscriber. Based on feedback from DSLAM, BRAS also dynamically adjusts the download rate limiter of the subscriber (Figure 3(a)); otherwise, precious bandwidth capacity in aggregation layer would be wasted since DSLAM would eventually drop extra packets. IETF developed a dedicated Access Node Control Protocol (ANCP) protocol for this kind of direct interaction between nodes (Wadhwa&Moisand, 2010).

We could move all state control up to a centralized controller, as demand by current SDN idea: shown in Figure 3(b), DSLAM reports to the controller about physical link capacity; based on the reports, controller configures the states in BRAS. The control loop then becomes a triangular path, which would significantly lower the efficiency due to the prolonged loop duration.

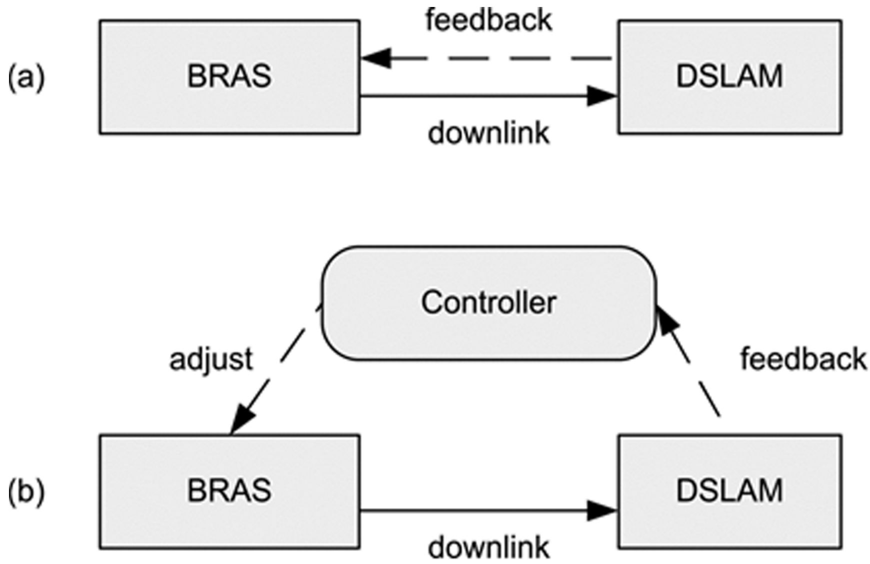
Another solid example is OLT-ONU uplink control. OLT controls the upload bandwidth allocation to different ONUs; the control cycle is normally just a few mille-seconds. If we move all state control up to a controller, the additional network latency would also significantly lower the efficiency.

Instantaneous Decision Inside Nodes

Of the devices we analyzed, it is a common practice to share packet buffers among several ports in the same interface card. It is well known that sometimes there could be flash-crowd traffic in a specific port; statically assigns fixed-size buffers to ports would lower the ability to handle such a flash-crowd. Most designs keep a public buffer space and there is a component which dynamically allocates a portion of the buffer to a demanding port.

Given the time-scale of packets arrival pattern variation, it is impossible to move this control state management to a centralized controller.

Figure 3. What if there is a controller?



WHAT IS THE RIGHT DEVICE ABSTRACTION?

The simple ''match-action'' abstraction of OpenFlow might be sufficient for data-center and enterprise networks, while we do found some problems.

Flaws of the ''Match-action'' Abstraction

Active Components?

The ''match-action'' abstraction is basically a passive model: actions are driven by incoming flow packets. But what if there are active components, which are not directly triggered by packets?

Scheduler is such an active component. Multi-tier hierarchical packet scheduler is a central part of QoS capability. Shown in Figure 4 is a typical scheduler for a port with 3 tiers: at the first level, the available bandwidth credits are allocated among subscriber groups; at the second level, credits of the group are allocated among subscribers; at the third level, credits of the subscriber are allocated among flow queues. The allocation algorithms could be Strict Priority (SP), Weighted Round Robin (WRR), Committed-Information-Rate/Peak-Information-Rate (CIR/PIR) etc.

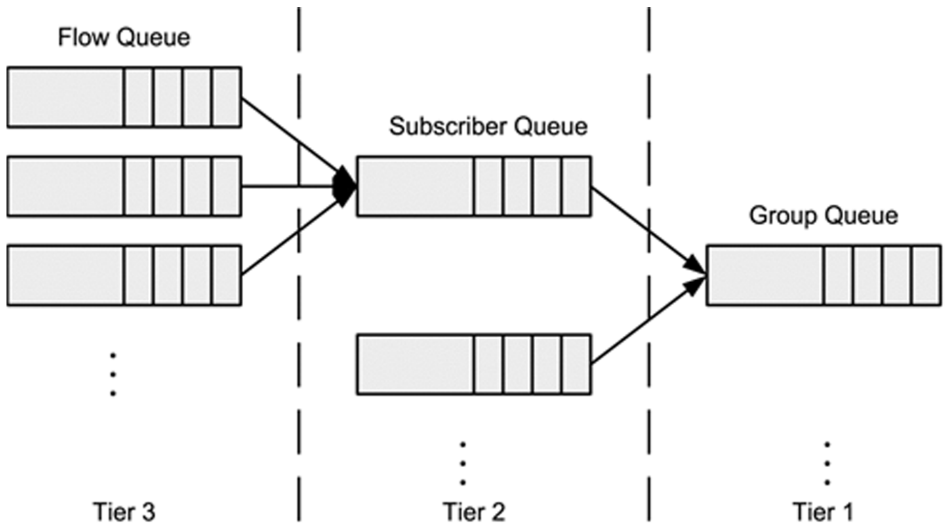
It is hard for ''match-action'' abstraction to model a complicated scheduler. The current OpenFlow specification only contains a very simple queue structure with CIR/PIR support. Algorithms like SP and WRR involve mutual relationships among flows, hence hard to be encoded in a ``match-action'' fashion.

System Attribute?

Another problem is that the ``match'' abstraction focuses only on the packet headers. The *METADATA* field in match structure is still attached with flows. There are some action decisions that depend not only on flow headers, but also on system state attributes.

Meter is such a case. The current OpenFlow specification contains a simple meter structure with a band table; each band defines a rate and the way to process the packets such as drop or mark colors. In access network devices, complex meters are not uncommon: their processing

Figure 4. A three-tier scheduler



logic could rely on buffer states, such as the occupation ratio.

It is hard for ''match-action'' abstraction to model system attributes. Flow structure is common to all devices, while system attributes are implementation-dependent: add them to ''match-action'' abstraction would inevitably compromise the uniform abstraction idea.

Abstraction vs. Implementation

An interesting reality in current SDN is the tight couple between abstraction and implementation. To be more specific, ''match-action'' is now not just an abstraction: almost all OpenFlow devices are implemented by TCAM matching tables. The questions are: is this really a MUST (in access networks)? Can we decouple implementation from abstraction?

First of all, ISPs have invested billions of dollars in existing networks. It is more reasonable to add SDN capability to existing devices, if possible, compared with replacing all of them with TCAM-based new devices.

Second, even for new access network deployments, pure TCAM-based implementations may not be the best choice. The commercial switching chips have been developed for tens of years; their implementation of different logic components (*e.g.*, classifier, Meter) is mature and cheaper than the TCAM-based version. Also a side-effect of identical hardware implementations is the loss of vendor differentiation.

RELATED EFFORTS

Neither centralized control nor uniform abstraction is a new adventure in network management. There are earlier efforts such as SNMP (Case&Mundy, 2002) and COPS-PR (Chan&Seligson, 2001). They failed to achieve the original goal what they were designed to, where we could learn some lessons from. A more recent effort is NETCONF/YANG (Enns&Bjorklund, 2011; Bjorklund, 2010), which we consider a potential supportive foundation to SDN abstraction in access networks.

SNMP and COPS-PR

SNMP

The IETF developed Simple Network Management Protocol (SNMP) in the late 1980s; it was supposed to be an "Internet-standard protocol for managing devices on IP networks". In typical SNMP settings, a Network Management System (NMS) monitors and/or manages a group of devices on a network. Each managed device has an agent which reports information via SNMP to the manager. Configuration states are SNMP management variables in both manager and devices. State control is done by modifying these variables.

SNMP was very popular, not being used to configure devices, but being used for network monitoring. The major reason is that it is not friendly to configuration task: Command Line Interfaces (CLI) is text-based, while SNMP is BER-encoded (Schoenwaelder, 2003).

COPS-PR

Another important proposal is Common Open Policy Service for Policy Provisioning (COPS-PR) (Chan&Seligson, 2001; Cohen&Herzog, 2000). There is a Policy Decision Point (PDP) in a dedicated server, and each device is a Policy Enforcement Point (PEP). COPS-PR supports two common models for policy control: Outsourcing and Configuration. The Outsourcing model is reactive as PEP requires an instantaneous policy decision when an event occurred (*e.g.*, flow in); the Configuration model is proactive as the PDP may proactively configures the PEP reacting to external events.

The COPS-PR proposal was also given up by operators mainly because ''its use of binary encoding (BER) for management data makes it difficult to develop automated scripts for simple configuration management tasks in most text-based scripting languages''(Schoenwaelder, 2003).

NETCONF

CLI has a bad nature: the content and formatting of output was prone to change in unpredictable ways. The unfulfilled requirement of centralized management led to the creation of NETCONF (Enns&Bjorklund, 2011).

Shown in Figure 5 is the NETCONF architecture. The NETCONF protocol uses an Extensible Markup Language (XML) based data encoding and a simple Remote Procedure Call (RPC) layer for the configuration data as well as the protocol messages. The RPC-based design allows the device to expose a full, formal API. The XML-based design reduces implementation costs and allows timely access to new features.

YANG (Bjorklund, 2010) is a ''human-friendly'' structured modeling language for defining the semantics of operational/configuration data, notifications, and operations for NETCONF. It could model the hierarchical organization of configuration data as a tree.

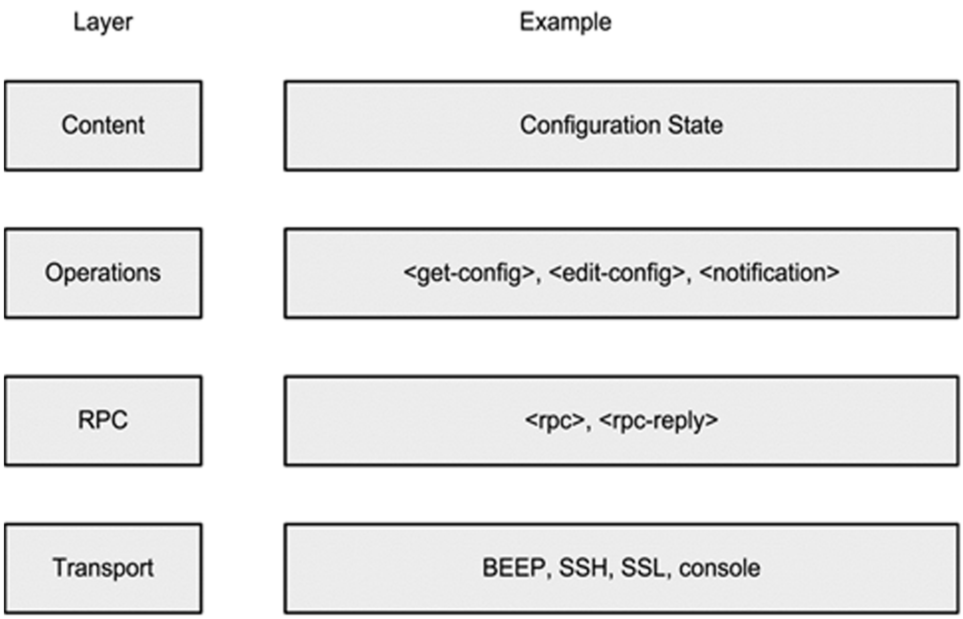
OBJECT-ORIENTED ABSTRACTION

Basic Ideas

Regarding challenges in access networks, we make some compromise to the pure-SDN philosophy.

- **Centralized Policy/State Control:** For Control layer, the separated and centralized control platform handles most states and all polices; it means that in certain scenario, it could delegate state control to devices while only control the policy.
- **Uniform Object-Oriented Abstraction:** For Infrastructure layer, devices of the forwarding plane are modeled by Click-like object-oriented abstraction;

Figure 5. NETCONF architecture



Click-like Model

Compared with the pipelined and table-driven OpenFlow programming model, Click-like *Software-defined* model (Kohler&Morris, 2000, pp.263-297) is much more friendly to network management programmer. Click-like model is object-oriented: define functional module classes; each device is a connected graph of functional class instances; instances are connected via either push or pull primitives; by configuring the properties of each instance, each flow takes a path through the whole graph.

The advantages of this object-oriented abstraction is: (1) it has a mapping relationship with existing hardware logic implementations, (2) it is intuitive to programmers, and (3) it makes capability extension, such as vendor specific feature, much easier.

Such an object-oriented abstraction could be expressed in any structured data models like YANG, and could be carried over expressive protocols such as NETCONF.

An Illustrative Example

In this example, we demonstrate that the ingress (to device) flow processing path of DSL subscribers in a BRAS can be modeled in an object-oriented abstraction. For example, we model scheduler as a functional class with properties shown in Table 2.

With just 9 functional classes, we can construct the whole configuration: Classifier, Meter, Marker, Dropper, Queue, Scheduler, DataPath, PushTag and PopTag. There are also three parameter classes: ClassifierEntry, Filter, MeterBand.

The scenario is: (1) there are two subscribers A and B, coming from the same DSLAM

Table 2. Scheduler class

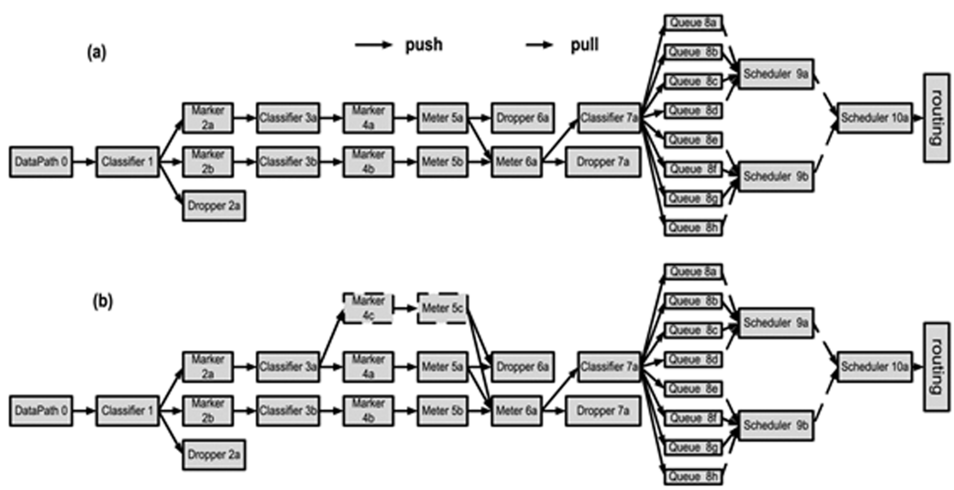
Property	Value
ID	Instance ID.
Next	The next functional Element ID/Null.
Method	Strict/WRR.
CIR	Commit Information Rate.
PIR	Peak Information Rate (Optional).
Priority	If strict priority scheduling.
Weight	If weighted round robin scheduling.

to a BRAS interface, and controller sets their upload/download bandwidth limitations respectively; (2) subscriber A requests for a BoD service after a while.

Shown in Figure 6(a) is the initial configuration. Taking subscribers A's path as the illustration.

- Incoming packets first get into Classifier 1; packets' metadata field belongs to subscribers A is marked in Marker 2a with A's subscriber ID.
- All packets of subscribers A are treated as one flow; hence they just pass through Classifier 3a.
- Packets are marked with the same Flow ID in Marker 4a and fed into Meter 5a for subscriber level control; over-limit packets are dropped by Dropper 6a.
- Packets from both subscriber A and B get into Meter 6a for port level rate limitation; over-limit packets are dropped by Dropper 7a.
- Subscriber A's packets are classified into different queues based on Differentiated

Figure 6. Ingress policy example



Services Code Point (DSCP) bits; the schedulers pull the packets to the routing fabric.

After the BoD service is added to subscriber A, controller calls APIs to add Marker 4c and Meter 5c into the graph, as shown in Figure 6(b):

- In Classifier 3a, BoD packets are marked a different Flow ID in Marker 4c;
- The BoD flow is fed into Meter 5c for BoD rate limitation.

DISCUSSIONS

We notice similar efforts in the Application layer: the Pyretic project builds a high-level programming model over underlying OpenFlow abstraction (Monsanto&Reich, 2013). While our position is that: it is possible to adopt object-oriented abstraction as the southbound API.

We are also not the only one noticed the potential of NETCONF/YANG in SDN. Deutsche Telekom's TeraStream approach has already adopted them as the major southbound interface (Clauberg&Millroth, 2013), while its technical details are unknown.

Network Function Virtualization (NFV) is a major trend from 2013 (Chiosi 2012; Martins 2014). Also notice that the current OpenFlow community focus on routing related issues, NFV propose to use industry standard high volume servers, switches and storage to replace the closed-form boxes (i.e., firewalls, accelerators, DPI). The realization of NFV would make the network really software-defined.

ACKNOWLEDGMENT

The authors would like to thank anonymous reviewers for their valuable comments. This work is partially supported by "National Natural Science Foundation of China (No. 61202107, No. 61100220, No. 61202303)", by "National High Technology Research and Development Program of China (863 Program No. 2014AA01A702)" and by the "Fundamental Research Funds for the Central Universities". Chen Tian is the corresponding author.

REFERENCES

Bjorklund, M. (2010). *YANG-A data modeling language for the Network Configuration Protocol*. NETCONF.

Canini, M., Venzano, D., Peresini, P., Kostic, D., & Rexford, J. (2012, April). A NICE Way to Test OpenFlow Applications. In NSDI (pp. 127-140).

Casado, M., Freedman, M. J., Pettit, J., Luo, J., McKeown, N., & Shenker, S. (2007). Ethane: Taking control of the enterprise. *Computer Communication Review*, *37*(4), 1–12. doi:10.1145/1282427.1282382

Case, J., Mundy, R., Partain, D., & Stewart, B. (2002). Introduction and applicability statements for internet standard management framework.

Chan, K., Seligson, J., Durham, D., Gai, S., McCloghrie, K., Herzog, S., & Smith, A. (2001). *COPS usage for policy provisioning*. COPS-PR.

Chiosi, M., Clarke, D., Feger, J., Cui, C., Benitez, J., Michel, U., . . . Sen, P. (2012, October). Network functions virtualisation: An introduction, benefits, enablers, challenges and call for action. In SDN and OpenFlow World Congress

Clauberg, A., & Millroth, H. (2013). SDN in DT TreaStream. *Journal of Proc. Open Networking Summit.*, from http://opennetsummit.org/pdf/2013/presentations.html

Durham, D., Boyle, J., Cohen, R., Herzog, S., Rajan, R., & Sastry, A. (2000). The COPS (common open policy service) protocol.

Elby, S. (2011). *Software defined networks: A carrier perspective*. Open Networking Summit.

End to End Architecture Working Group. TR-101 Issue 2: Migration to Ethernet-Based Broadband Aggregation.

End to End Architecture Working Group. TR-200: Using EPON Access in the Context of TR-101.

Enns, R., Bjorklund, M., & Schoenwaelder, J. (2011). NETCONF configuration protocol. *Network*.

Gopal, I. (2013). Introduction to OpenDaylight. *Journal of Proc. Open Networking Summit.*, from http://opennetsummit.org/pdf/2013/presentations.html

Gude, N., Koponen, T., Pettit, J., Pfaff, B., Casado, M., McKeown, N., & Shenker, S. (2008). NOX: Towards an operating system for networks. *Computer Communication Review*, *38*(3), 105–110. doi:10.1145/1384609.1384625

Jain, S., Kumar, A., Mandal, S., Ong, J., Poutievski, L., Singh, A., & Vahdat, A. et al. (2013, August). B4: Experience with a globally-deployed software defined WAN. In *Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM* (pp. 3-14). ACM. doi:10.1145/2486001.2486019

Kohler, E., Morris, R., Chen, B., Jannotti, J., & Kaashoek, M. F. (2000). The Click modular router. [TOCS]. *ACM Transactions on Computer Systems*, *18*(3), 263–297. doi:10.1145/354871.354874

Koponen, T., Casado, M., Gude, N., Stribling, J., Poutievski, L., Zhu, M., . . . Shenker, S. (2010, October). Onix: A Distributed Control Platform for Large-scale Production Networks. In OSDI (Vol. 10, pp. 1-6).

Martins, J., Ahmed, M., Raiciu, C., Olteanu, V., Honda, M., Bifulco, R., & Huici, F. (2014, April). ClickOS and the art of network function virtualization. In11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14). Seattle, WA: USENIX Association (pp. 459-473).

Matsumoto, C. (2012). Carrier Ethernet Has a Job for SDN. From http://www. lightreading.com/software-defined-networking/carrier-ethernet-has-a-job-for-sdn/240143102.html

Mcdysan, D. (2013). Software defined networking opportunities for transport. *Communications Magazine, IEEE*, *51*(3), 28–31. doi:10.1109/MCOM.2013.6476862

McGarry, M. P., & Reisslein, M. (2012). Investigation of the DBA algorithm design space for EPONs. *Journal of Lightwave Technology*, *30*(14), 2271–2280. doi:10.1109/JLT.2012.2196023

McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., & Turner, J. et al. (2008). OpenFlow: Enabling innovation in campus networks. *Computer Communication Review*, *38*(2), 69–74. doi:10.1145/1355734.1355746

Monsanto, C., Reich, J., Foster, N., Rexford, J., & Walker, D. (2013, April). *Composing Software Defined Networks* (pp. 1–13). NSDI.

Schoenwaelder, J. (2003). Overview of the 2002 IAB network management workshop. Ng, E. Maestro: A System for Scalable OpenFlow Control.

Sezer, S., Scott-Hayward, S., Chouhan, P. K., Fraser, B., Lake, D., Finnegan, J., . . . Rao, N. (2013). Are we ready for SDN? Implementation challenges for software-defined networks. Communications Magazine, IEEE,51(7).

Shenker, S., Casado, M., Koponen, T., & McKeown, N. (2011). *The future of networking, and the past of protocols*. Open Networking Summit.

Sherwood, R., Gibb, G., Yap, K. K., Appenzeller, G., Casado, M., McKeown, N., & Parulkar, G. (2009). Flowvisor: A network virtualization layer. OpenFlow Switch Consortium, Tech. Rep.

Wadhwa, S., Moisand, J., Haag, T., Voigt, N., & T Taylor, E. (2011). Protocol for Access Node Control Mechanism in Broadband Networks. *draft-ietf-ancp-protocol-17. txt, April*.

Yu, M., Jose, L., & Miao, R. (2013, April). Vol. 13, pp. 29–42). Software Defined Traffic Measurement with OpenSketch. In NSDI.