SDN-enabled Application-aware networking for Datacenter networks

Muzzamil Aziz*, Amirreza FazelyH.†, Giada Landi‡, Domenico Gallico§, Kostas Christodoulopoulos¶, Philipp Wieder‖
*Gesellschaft für wissenschaftliche Datenverarbeitung mbH (GWDG), Göttingen
Email:muzzamil.aziz@gwdg.de
†Gesellschaft für wissenschaftliche Datenverarbeitung mbH (GWDG), Göttingen
Email: amirreza.fazely.hamedani@gwdg.de
‡Nextworks, PISA, Italy
Email: g.landi@nextworks.it
§Interoute SpA, Italy
Email: domenico.gallico@interoute.com
¶Communication Networks Laboratory of the Computer Engineering and Informatics Department, University of Patras
Email: kchristodou@ceid.upatras.gr
‖Gesellschaft für wissenschaftliche Datenverarbeitung mbH (GWDG), Göttingen
Email: philipp.wieder@gwdg.de

***Abstract*—Software-Defined Networking (SDN) is a networking paradigm that decouples the control plane of a network from its forwarding plane and offers programmability of the data plane devices to manage and control the ongoing traffic flows. This paper presents the control plane architecture of a Datacentre network (DCN) and its operational services being developed for NEPHELEs optical network infrastructure. The heart of the proposed control plane overlay is an OpenDaylight SDN controller, which along-with its north and south-bound interfaces bridges the gap between the cloud applications and the DC network configuration at the data plane, in order to automatically adjust the underlying network to the Quality of Service (QoS) requirements at the application level. An application based traffic shaping use case is presented as a proof of concept of application aware networking in SDN-enabled data center networks.**

I. Introduction

NEPHELE is a multidisciplinary European research project that aims to develop a dynamic optical network infrastructure for highly scalable, disaggregated datacentres. The scalability of a traditional 'fat tree' datacentre architecture is under question today due to its non- linear expansion to cope with the steady increase of traffic [1] in cloud and other content distribution applications etc. To tackle this problem, NEPHELE proposes a novel DCN architecture that aims to overcome the limitations of traditional DC design and drastically reduce the cost and power consumption of the cloud DCs.

At its control plane level, NEPHELE utilizes the strength of the emerging SDN paradigm to support an innovative phenomena of application aware networking in cloud data centers. On one side, it enables the efficient allocation of network resources in DCNs and, on the other side, it ensures the QoS guarantees required by the cloud applications running over the virtual infrastructure and performs real-time optimizations based on the global traffic matrices generated by those applications.

A. NEPHELE Control Plane Architecture

The objectives of the NEPHELE SDN control plane architecture are twofold: first, maintaining a topological view of the underlying existing data plane network and providing resource optimizations on real-time basis. Second, collaborating and providing an interface to the cloud orchestration for the deployment and configuration of virtualized networks. Hence, the proposed architecture is composed of two major components:

- Network control framework: It is the in charge of coordinate resource allocation in the DCN through an SDN controller which interacts with the data plane devices using OpenFlow protocol.
- Cloud orchestration framework: It is responsible for coordinating and orchestrating the usage of compute, storage and network resources in the NEPHELE DC.

Figure 1 presents a high level view of NEPHELE SDN control plane architecture, where OpenDaylight [3] has been introduced as an NEPHELE SDN controller. At its northbound interface, the SDN controller interacts with the OpenStack as a cloud orchestration platform hosting cloud applications. Here, the interaction is made possible via a REST messaging interface. From the implementation point of view, the configurations and provisioning of the virtual resources by OpenStack are carried out by its orchestrator component called Heat. It is mainly responsible to coordinate and synchronize the deployment of the different kinds of DC resources (network, computing, storage). Furthermore, it is also responsible of managing the real-time and on-demand requests of scaling up or down the network resources. Other OpenStack components such as Nova, Swift (also Cinder and Glance) and Neutron are responsible for configuring compute, storage and network resources respectively.

At the southbound interface, the SDN controller interacts with the NEPHELE switching devices, such as TORs, POD

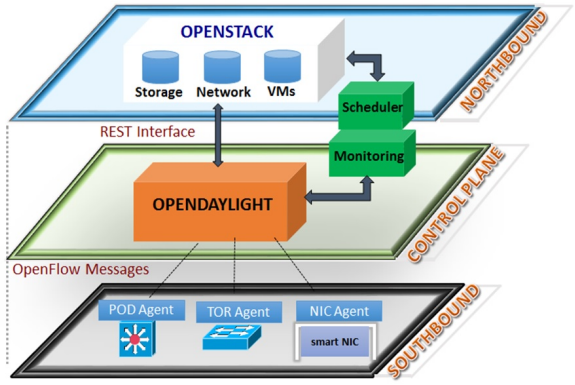

Fig. 1. High level view of NEPHELE SDN Control Plane

switches and NICs. The interaction here is made via the device specific OpenFlow agents, which behave like proxies to actual devices at the data plane level. The interaction with the data plane optical devices pose several challenges to the standardized implementation of OpenFlow agents. A standardized OpenFlow agent based on OpenFlow version-1.3 supports electrical Ethernet switches only, hence, contains no ability to manage NEPHELE specific optical resources. Therefore, the proposed NEPHELE specific OpenFlow agents need to introduce extensions at three different levels:

- Advertisement of active ports, switching capabilities and available wavelengths of the data plane devices.
- Operational configuration of the devices from the Controller (e.g. adding flows, creating cross-connections)
- Asynchronous notifications from the data plane to the controller and retrieval of counters from the controller to the data plane.

B. NEPHELE control plane features and applications

The heart of the NEPHELE control plane architecture is the SDN controller, which hosts the necessary DCN core services and programming applications required for the seamless operation of the whole NEPHELE architecture. Figure 2 presents a functional view of the NEPHELE SDN controller. At the bottom level, the NEPHELE DCN southbound drivers group all the components that manage the interaction with the data-plane forwarding devices. The intermediate level, namely the NEPHELE DCN core services, consists of the components providing a set of core services of an SDN-enabled network. The top level comprises of the specialized NEPHELE services which are in charge of managing the interaction with the orchestrator.

The services implemented in the NEPHELE SDN aplications are exposed by REST APIs over the HTTP protocol at the controller's NBI, modeling resources which are manipulated through Create, Read, Update, Delete actions. The providers of the services implement an HTTP server, while the consumers implement an HTTP client. The messages exchanged on the NBI are HTTP requests and replies, where the URL of the HTTP message identifies the resource, while its attributes are carried in the message body in a JSON format. The information models associated to these resources are described through the YANG language. In the following a brief description of the main NEPHELE SDN applications and controller core services, together with the REST APIs is provided:

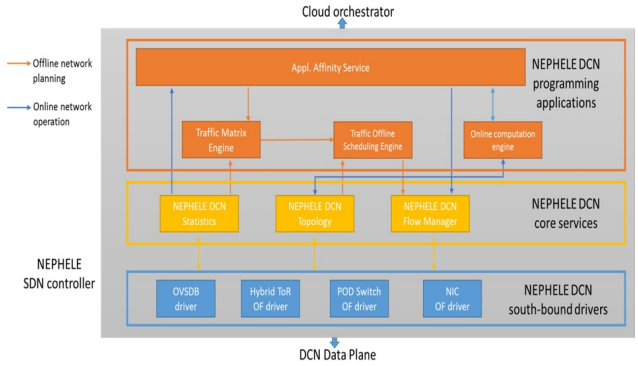

Fig. 2. NEPHELE SDN Controller functional components

- Application Affinity Service: provides the highest entry point in the DCN control plane which is invoked by the Cloud Orchestrator (OpenStack) to create, modify or destroy network connections when provisioning, instantiating or removing virtual networks for cloud applications. The list of APIs is specified in Table I. It triggers the online computation engine for obtaining a path and reserves the required resources. It also notifies the traffic matrix engine about the expected application profile to enable an application-aware and global DCN optimization in the medium-long term.

HTTP Method	URI	Description
POST	/affinity/connection	Establish a new DCN connection.
GET	/affinity/connections	Get the list of all the connections.
GET	/affinity/connection/connectionID	Get the details of an existing connection.
DELETE	/affinity/connection/connectionID	Terminate an existing connection.

TABLE I. APPLICATION AFFINITY SERVICE REST APIS

- NEPHELE DCN Topology Manager: provides a read-only service to query the current topology of the DCN, in terms of nodes, ports and links, together with their up-to-date resource availability and constraints. Table II refers to the API supported by DCN Topology Manager.

HTTP Method	URI	Description
GET	/toplogymanager/topology	Get the current topology.

TABLE II. DCN TOPOLOGY MANAGER REST APIS

- NEPHELE DCN Flow Manager: provides mechanisms to add, remove, modify and query flows in the DCN data plane devices. Table III refers to the APIs supported by DCN Flow Manager.
- NEPHELE DCN Statistics Manager: provides a read-only service (Table IV) to query the monitoring

HTTP Method	URI	Description
POST	/flowmanager/flow	Create a new flow.
GET	/flowmanager/flows/nodeID	Get the list of all the flows configured in a node.
GET	/flowmanager/flow/nodeID/flowID	Get the details of an existing flow.
DELETE	/flowmanager/flow/nodeID/flowID	Remove an existing flow.

TABLE III. DCN FLOW MANAGER REST APIS

data collected from the network (e.g. counters) or to receive notifications about network failures through subscribe/notification procedures.

HTTP Method	URI	Description
GET	/statisticsmanager/nodeID	Get the counters for the given node.

TABLE IV. DCN STATISTICS MANAGER REST APIS

- Traffic Matrix Engine: provides mechanisms to build the traffic matrix based on monitoring data or the expected application behavior in terms of traffic profile. This service operates continuously to optimize the usage of the network resources and exposes a management interface to retrieve the computed traffic matrixes or to add/remove new traffic profiles.
- Traffic Offline Scheduling Engine: provides mechanisms to allocate resources on the DCN in the medium/long term, optimizing the usage of the whole network based on the traffic matrix computed by the related engine (Table V).

HTTP Method	URI	Description
POST	/offlinescheduling/networkallocation	Computes a new network resource allocation.
GET	/offlinescheduling/networkallocation /networkallocationID	Retrieves a network resource allocation already computed.

TABLE V. OFFLINE SCHEDDULING SERVICE REST APIS

- Online Computation Engine: provides mechanisms to compute network paths between the DC servers, required to establish connections for supporting single cloud services (Table VI). It operates on-demand, triggered by the Application Affinity Service in case of new service requests or fast recovery of failed services.

HTTP Method	URI	Description
POST	/onlinecomputation/path	Commit the computation of a new connection path.
GET	/onlinecomputation/path/pathID	Get the computed path.

TABLE VI. ONLINE COMPUTATION SERVICE REST APIS

C. Control Plane Hierarchical Models

The maximum size of the NEPHELE DC network is targeted as 400 PODs, 1600 TORs and 6400 NICs at the Innovation Zones for a total number of 8400 devices. Considering the number of virtual machines as 50, each serving around 500 flows, sum up to the total number of about 160 million flows at a time. Controlling such a big network with a single centralized controller can pose several scalability concerns to the network. Thus, the possibility to adopt a distributed deployment model should be evaluated, taking into account the benefits of a better load balancing but also, on the other hand, the additional complexity due to the required coordination between SDN controllers. Some potential architectural models proposed for the control plane architecture are as follows (figure 3):

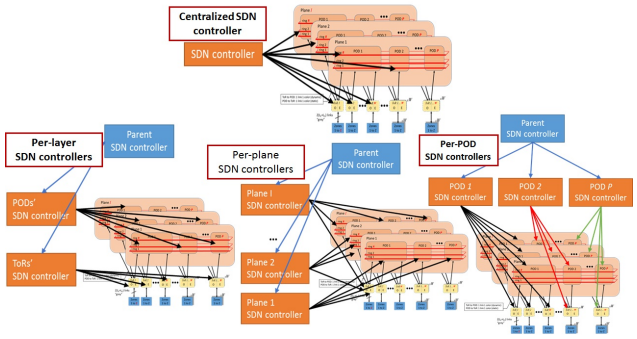

Fig. 3. Different hierarchical models for SDN controllers network

- Per-layer SDN controller: This model recommends an additional dedicated SDN controller for each type of switching devices at the data plane level i.e., a centralized parent controller with two child controllers for ToR and POD switches. It is worthy to mention that the model fits well with the classical DCN approaches exposing the separation of leaf and core, however, there is a possibility of a scalability concern in terms of large traffic burden at the TORs' controller compared to the traffic burden on PODs' controller.
- Per-plane SDN controller: This model proposes additional child controllers for each single plane. This modeling approach at least solves the asymmetric distribution of traffic burden and devices to different Per-layer controllers. However, the model does not reflect the hierarchy of the traffic flow i.e., the logical coherence of the flows can not be met with the physically distributed controllers between the planes.
- Per-POD SDN controller: This hierarchical model is considered as a preferable choice for NEPHELE DCN because of its fair distribution of traffic load among the child controllers by the parent controller. Moreover, the partition of the network also resembles the logical distribution of the flows in this particular design.

D. Application aware networking

The programmability of networking devices has made the interaction between the network and the applications possible. The outcome of which has enhanced the network capabilities to collect the application requirements and perform resource allocations and optimizations accordingly. Similarly, one of the objectives of the NEPHELE control and orchestration framework is to orchestrate the application requirements to the SDN controller and to provide a monitoring service that can compare the real time performances to what has been

decided, promised or expected by the applications initially. The following section presents a business use case of application aware networking in SDN-enabled DCNs. A similar use case with the example of Youtube streaming application is presented here [6].

1) Use case - Application based traffic shaping: Traffic shaping is a known traffic management technique used by the network administrators for traffic optimizations in the network. This section presents how an SDN control plane solution can help the network administrators to set traffic shaping rules in the network with just few number of clicks. A prototype application is developed for campus data center networks in this regard. The end users of a campus network can be broadly categorized as employees, students and guests. The application GUI lets the employees record the timings of their important project meetings or conference calls in a calendar. Based on this calendar data, the application allows the network administrators to set the traffic shaping rules in order to offload some of the traffic from the students and guests accounts in the event of important employees meetings. Currently, the application supports two different operations to offload network traffic. First, the reduction in bandwidth of a given link, and second, as to deny access to certain type of application traffic. Once the rules are saved by the network administrator, they are automatically converted into REST API calls to insert corresponding static flows in the OpenFlow switches.

The above mentioned shaping operations are possible with the help of RESTCONF APIs (/restconf/config/) [4] of OpenDaylight (Lithium) SDN controller. As the NEPHELE data plane devices are still being developed, the application prototype is tested with OpenVSwitch and OpenFlow 1.3 [2]. The testbed topology is created with Mininet network emulator, consisting of 30 client machines (10 Employee, 10 Students and 10 Guest) and 3 server hosts (HTTP, UDP and TCP). For prototyping, a simple DC design with two level switches in tree-leaf architecture is considered, allocating the clients' network to one leaf of switch and the servers' network to another leaf.

Figures 4 and 5 depict the network performance comparison of a TCP session in a normal and restricted evaluation modes. Out of 30 clients machines sending concurrent traffic requests (10 TCP, 10 UDP and 10 HTTP), the performance results of a single machine is computed with the help of iperf [5] network utility. The figures show that the available network bandwidth in restricted mode is much bigger than the one in the normal mode. This is because of the reason that the link bandwidth for guests and students accounts is reduced to 1 MB only in the restricted evaluation mode compared to 20 MB link in the normal mode. Similarly, figures 6 and 7 present the results of a UDP session in the same manner as of TCP. The overall behavior of the network in this case is same as TCP, although there is no much difference between the network bandwidth of normal and restricted evaluation mode.

II. Conclusion

The application aware networking is an advanced network provisioning service of DCNs that includes the collection of application requirements from the cloud orchestrator and performs two activities in return: 1) To ensure the provisioning of demanded networking resources at the data plane level 2) To perform real-time traffic optimization in the network in order to enhance the QoS of the running applications to the level of agreed SLAs. This paper presents the SDN based control plane architecture of a NEPHELE European project that aims to increase the efficiency of the cloud DCNs with the introduction of a hybrid electronic-optical architecture for DCNs. The paper explains how the above identified activities of an application aware monitoring service are performed with the proposed control plane for NEPHELE DCN.

Fig. 4. Network Performance of a TCP session in normal evaluation mode

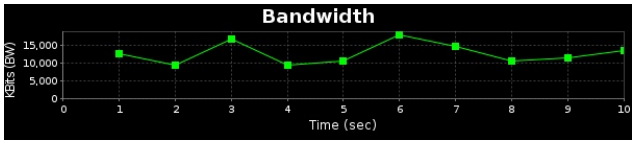

Fig. 5. Network Performance of a TCP session in restricted evaluation mode

Acknowledgment

This work has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No 645212 (NEPHELE).

References

[1] Cisco, *Cisco Global Cloud Index: Forecast and Methodology, 2014-2019*, 2015.

[2] Open Networking Foundation, OpenFlow Switch Specification. Version 1.3.0, ONF TS-006, June 2012.

[3] S. Rao. SDN Series Part Six: OpenDaylight, the Most Documented Controller.

[4] A. Bierman and M. Bjorklund and K. Watsen, *RESTCONF Protocol - draft-bierman-netconf-restconf-02*, IETF 88, November2013.

[5] A. Tirumala and F. Qin and J. Dugan and J. Ferguson and K. Gibbs, *The TCP/UDP bandwidth measurement tool*, May 2005, http://NANAR.net

[6] M. Jarschel and F. Wamser and T. Hhn and T. Zinner and P. Tran-Gia, *SDN based Application Aware Networking on the Example of YouTube Video Streaming*

Fig. 6. Network Performance of a UDP session in normal evaluation mode

Fig. 7. Network Performance of a UDP session in restricted evaluation mode