Chapter 83
Design of General SDN Controller System Framework for Multi-domain Heterogeneous Networks

Wenxiao Li, Jianhua Zhao, Huicong Fan, Shijia Zhu, Wandi Liang, Hongguang Yu, and Peng Lin

Abstract The integration of two technologies, Cloud Computing and Software Defined Network (SDN) has placed cooperative orchestration between cloud and network into the focus of attention. However, with the continuous expansion of network scale, the data center would inevitably accommodate multiple SDN network devices which are produced from various manufacturers, and even these devices are associated with different SDN technologies. Therefore, application scenario with multi-domain heterogeneous SDN networks occurs. The interoperability and orchestration of cross-domain services, as well as unified management of devices from multiple domains, have become a challenge. At present, except for OpenFlow, the mainstream technologies for SDN implementation include segment routing technology based on MPLS (SR-MPLS) and that based on IPv6 (SRv6). This paper proposes a general SDN controller system framework. This framework is able to shield the discrepancies, which is caused by different SDN technologies, between northbound interfaces of SDN controllers from multiple manufacturers. Besides, it supports unified management of network resources to solve the difficulties of orchestration of cross-domain service in SDN network. Furthermore, this system framework follows the design principles of high cohesion and low coupling, and regard every adapter that manages one SDN domain with specific technology as one sub-module in the whole adapter module, hence with flexible expansion.

Keywords Software defined network · Segment routing technology · Orchestration of cross-domain service

W. Li · J. Zhao · H. Fan · S. Zhu · W. Liang · H. Yu
Institute of Economy and Technology, State Grid Hebei Electric Power Co, Ltd., Hebei 050023, China

P. Lin (✉)
Beijing VectInfo Technologies Co., Ltd, Beijing 100088, China
e-mail: linpeng@vectinfo.com

S. Li (ed.), *Computational and Experimental Simulations in Engineering*, Mechanisms and Machine Science 145, https://doi.org/10.1007/978-3-031-42987-3_83

83.1 Introduction

In recent years, with the development of cloud computing technology as well as the continuous expansion of scale of networks in cloud data centers, it has become a trend that cloud and network get converged. A few operators and even large enterprises have already possessed their own technologies, which is about the integration of cloud and network, with various degrees of development. This technology is becoming the foundation and key of the novel communication infrastructure [1]. SDN is an innovative network architecture technology, which is emerging in recent years. Due to the flexibility, programmability and scalability of this technology, it is capable to solve the problems faced during cloud computing in large scale networks. The cloud computing network based on SDN will achieve good operation effect in practice and meet demand for the development of cloud computing [2]. However, as the continuous growth of the scope of cloud resource pool and SDN devices deployment, the situation of multi-vendor SDN environment would inevitably emerge, resulting in the demand for intercommunication, unified management and collaborative orchestration in heterogeneous SDN network [3]. In general, the network devices, which are controlled by one controller, from the same SDN domain usually are applied with the same SDN technology, also from the same manufacturer. In the environment of integration between cloud and network, as the scale of cloud computing expands, there may be deployed with multiple multi-vendor SDN products in the cloud resource pool with large scale in one data center [4]. In the scenario of multiple heterogeneous SDN network domains, with regards to devices in different domains, the technologies applied and manufacturers may be not exactly the same. Therefore, it is difficult for the management platform of cloud network in single data center to ensure interoperability and cooperative orchestration of the cross-domain services.

In addition to the current mainstream SDN technologies, OpenFlow and SR-MPLS, which had been applied, a new SDN technology called SRv6 has been realized in some devices. With respect to SRv6, not only the application scenarios and commercial value are now being exploring, but also the industrial standard is still in the draft stage. However, the strengths of SRv6 have been widely recognized by the industry [5]. At present, there have already been a few SDN implementation technologies as shown in Table 83.1. Though during the process of implementation in different SDN technologies different network resources, such as flow tables in OpenFlow and tunnels in SR-MPLS, are required to operate in actual service process, the format or meaning of parameters required by underlying devices are not exactly when orchestrating service. As a consequence, in multi-domain heterogeneous networks, when the service orchestration system in upper layer issues service configuration instructions to underlying devices from top to bottom, it is required to shield discrepancies leaded by not only different manufacturers but also different SDN techniques.

Table 83.1 The information of current mainstream SDN implementation technology

SDN implementation technology	Application scenario	Scope of service deployment	Main function
OpenFlow	Usually applied in local area network with small scale, not suitable network with large scale	Need to deploy service on all nodes included in service path	Realize division of packets from specific service by flow table, limit service bandwidth through queue
Segment routing based on MPLS(SR-MPLS)	Based on IPv4 applicable to wide area network with large scale	Only need to deploy service on head node and tail node of service path	Realize division of service packets and resource allocation for bandwidth though SR tunnel. Provide function of path centrally calculating
Segment routing based on IPv6(SRv6)	Based on IPv6 applicable to wide area network with large scale	Only need to deploy service on head node and tail node of service path	Define topology and forwarding behavior through segment policies. Provide differentiated forwarding paths and isolate resources for network slices. Provide three-layer programmable space to meet complex requirement of service

83.2 The Challenge of Multi-domain Heterogeneous SDN Network

China Telecom has independently developed the SDN collaborative orchestrator called SDN-O for IDC (Internet Data Center) in scenarios of traffic scheduling intelligently. It interacts with the interface of SDN controllers from multiple vendors to realize cooperative management of the controllers. At present, the SDN-O system has supported the access of SDN controllers from mainstream manufacturers such as Huawei, Cisco, Shanghai Bell, and H3C etc. The interface protocols it supporting include Restful and Netconf etc. China Unicom currently realizes the management of heterogeneous SDN network through self-developed system. Instead of the application, which is developed in upper layer, based on SDN controllers from manufacturers, the system is comparatively independent from the manufacturer's control system through directly connecting with their devices [6]. China Mobile also creatively proposes the concept of POD (Point of Delivery), and apply multi-POD networking structure technology [7]. The comparison of these solutions is shown in Table 83.2.

So far, for the management of SDN devices from different manufacturers, most enterprises have developed service orchestration management systems by themselves to directly manage network devices or SDN controllers. However, some problems still exist.

Table 83.2 Some solutions for management of multi-vendor SDN device from some operators

Operator	Solution
China telecom	Independently develop collaborative orchestrator, which interacts with SDN controllers from multiple vendors, to realize cooperative management of the controllers
China unicom	Independently develop network management system, which directly take charge of devices from vendors by protocols (such as Netconf), to shield interface differences
China mobile	Propose the concept of Point of Delivery (POD), which is centrally managed by a unified cloud management platform. Besides, define and unify the parameters contained in the interface specification to ensure the standardization of northbound interface of SDN controller from multiple manufacturers

- The management protocol of SDN devices produced by some manufacturers may be not available. In orchestration system, it is required to invoke the northbound interface of SDN controllers from these manufacturers to indirectly control network devices, which means that an independent module is needed to develop to interact or manage the SDN controller from one specific manufacturer. It leads to poor scalability in this system.
- The control protocols of network devices from different manufacturers are not exactly the same. when directly managing them, the driver framework provided by manufacturer is required to embed into the module, which is in charge of devices from this manufacturer in the system. As a result, the information exchange between these modules is complex.
- When applying different SDN technologies in service orchestration system, discrepancies between them embody in not only the information models but also the invocation order, entities, and parameters in service process. In upper layer, they are needed to distinguish.

83.3 General SDN Controller System Framework for Multi-domain Heterogeneous Networks

In actual project, from the macroscopic perspective, the layering of management of multi-domain heterogeneous SDN network is shown in Fig. 83.1. And the detail description for each specific layer is as follows:

1. High-Level Application Layer: Other system (such as cloud management platform), which is in the top, interacts with the SDN network management layer below. Through invoking the northbound interface of cross-platform SDN network management platform to proceed instructions issue and service configuration.
2. SDN Network Management Layer: It is mainly composed of a cross-platform SDN network management platform, which is divided into these following functional modules: service orchestration management module, configuration

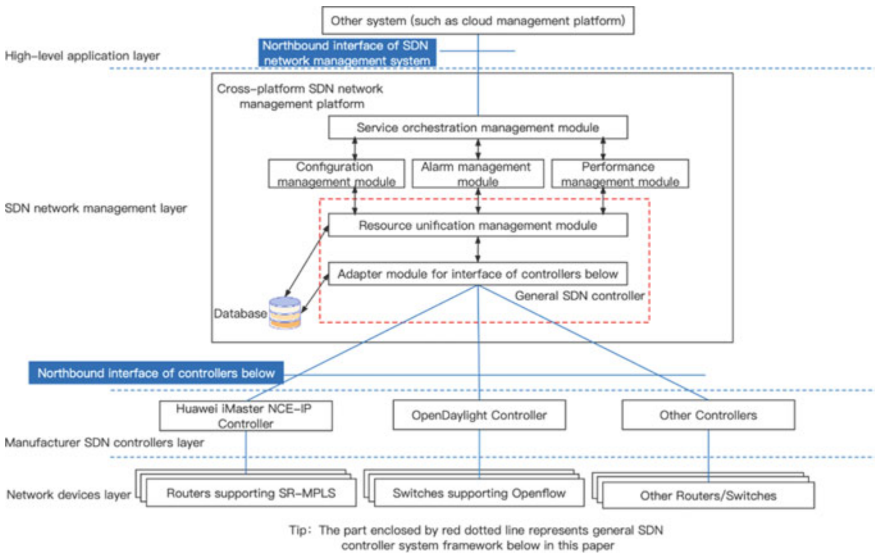
Other system (such as cloud management platform)
Northbound interface of SDN network management system
High-level application layer
Cross-platform SDN network management platform
Service orchestration management module
Configuration management module
Alarm management module
Performance management module
SDN network management layer
Resource unification management module
Adapter module for interface of controllers below
General SDN controller
Database
Northbound interface of controllers below
Huawei iMaster NCE-IP Controller
OpenDaylight Controller
Other Controllers
Manufacturer SDN controllers layer
Network devices layer
Routers supporting SR-MPLS
Switches supporting Openflow
Other Routers/Switches
Tip: The part enclosed by red dotted line represents general SDN controller system framework below in this paper

Fig. 83.1. The layering of system in multi-domain heterogeneous SDN network

management module, alarm management module, performance management module, resource unification management module and adapter module for interface of controllers below.

3. Manufacturer SDN Controllers Layer: The controller, which is a software with control function, is customized and developed by manufacturers for their respective SDN devices (routers and switches). It is utilized to control the flow table and routing switching strategy in devices as well as monitor device status. Usually, the controller developed by one manufacturer could only manage devices produced by the same manufacturer. Besides, it provides a northbound interface for invocation or secondary development for users. Nevertheless, normally it is not be able to manage devices from other manufacturers. (The controllers deployed in this scenario are Huawei iMaster NCE-IP and OpenDaylight.)
4. Network devices layer: Refers to the underlying network devices which are deployed with services. (In this scenario, the router (model Huawei NE8000 M1A) supports SR-MPLS, and SDN switches (model Pica P-5101) supports OpenFlow.)

Among the layers above, the general SDN controller system framework proposed in this paper is made up of the resource unification management module and adapter module in the SDN network management layer. The details inside the system are shown in Fig. 83.2.

The general SDN controller system is composed of two modules which are named resource unification management module and adapter module respectively.

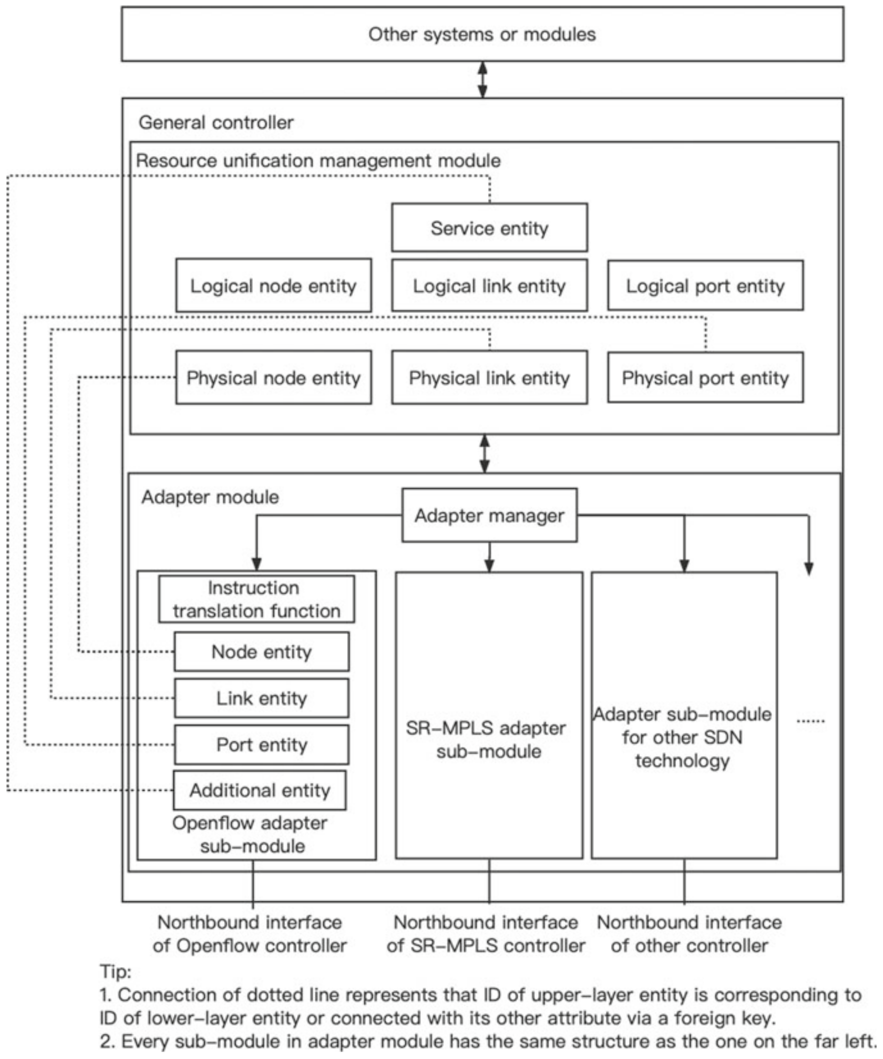
Other systems or modules
General controller
Resource unification management module
Service entity
Logical node entity
Logical link entity
Logical port entity
Physical node entity
Physical link entity
Physical port entity
Adapter module
Adapter manager
Instruction translation function
Node entity
Link entity
Port entity
Additional entity
Openflow adapter sub-module
SR-MPLS adapter sub-module
Adapter sub-module for other SDN technology
......
Northbound interface of Openflow controller
Northbound interface of SR-MPLS controller
Northbound interface of other controller
Tip:
1. Connection of dotted line represents that ID of upper-layer entity is corresponding to ID of lower-layer entity or connected with its other attribute via a foreign key.
2. Every sub-module in adapter module has the same structure as the one on the far left.

Fig. 83.2 The system frame diagram of general SDN controller

83.3.1 Resource Unification Management Module

The main function of the resource unification management module is to unify the heterogeneous network resources, which locates on different SDN domains in bottom layer, into seven entities for management. So that for modules or systems in upper layer, differences between these domains are eliminated when performing management. This module provides an interface, which is able to be invoked to realize interaction with other systems or modules, such as configuration management module mentioned above.

These seven entities managed by this module can be roughly classified into two categories. Entities, including physical nodes, physical ports and physical links, belongs to one category featuring with objective existence in network. All of them correspond to actual elements can be seen in concrete network. For instance, physical nodes correspond to switches or routers. The rest ones, namely logical nodes, logical ports, logical links and services, belong to the other category. They are abstract and invisible in network. One service consists of three other abstract entities. Therefore, they would change as addition, deletion, or modification of service they belong to.

83.3.2 Adapter Module

The adapter module provides adaptation function for northbound interfaces of different SDN controllers. It maps entities in resource unification management module to those in underlying network to ensure the transmission of correct service instructions to the corresponding SDN controller. This module is composed of an adapter manager and adapter sub-modules that representing different SDN implementation technologies.

The adapter manager, which is considered as the core of the whole adapter module, manages the operations of each sub-module in service process. It receives full command parameters from resource unification management module and decomposes it into the parameters required by each sub-module. Finally, it provides these parameters to each adapter sub-module respectively, as well as inform resource unification module above of the result.

Each adapter sub-module, which stands for a particular SDN implementation technology, possesses with the same partitioning of internal functionality. In each one, the functions are divided into node management, port management, link management, additional entities management and instruction translation.

The node entity in sub-module stores the extended attributes required by a physical device node under this particular SDN technology. It will have one-to-one correlation with a physical node in resource unification management module. So do the port entity and link entity.

Additional entities in sub-module represent the particular entities, which are related to high-level service, are contained in underlying device under this particular SDN technology. For example, in order to meet bandwidth requirement in one service, a queue entity is required to create and utilize in OpenFlow, while bandwidth information will be saved in SR tunnel entity in SR-MPLS. These entities would change as addition, deletion and modification of service, thus attached to service in some extent. Therefore, the foreign key of at least one of their attributes should correspond to ID of one high-level service.

The instruction translation section in sub-module stores the steps of operations performed by sub-module to execute instruction in service process. For example, invoke specific methods in northbound interface of a SDN controller, query required parameters in this sub-module, or create additional entities corresponding to service.

The general controller has advantages with satisfactory scalability from the perspective of design. When there is a requirement to add sub-modules corresponding to other SDN technologies, it only requires to append sub-modules in adapter module according to the same design principle.

83.4 Methodology

83.4.1 Information Model

The UML class diagram of this general controller is shown in Fig. 83.3. In resource unification management module, seven entities mentioned above are not be described here to avoid repetition. Among the newly added entities, IntradomainService represents a sub-service in one SDN domain, and PhysicalNetwork stands for a real physical network containing physical nodes and physical links.

In Fig. 83.3, the sub-modules contained in adapter module are uniformly designed according to the class diagram shown, and dotted line represents that these attributes are connected via a foreign key. In each adapter sub-module, the primary keys, namely ID properties, of nodes, ports and links are respectively to those of physical nodes, physical ports and physical links in resource unification management module through foreign keys, indicating that there always exists a one-to-one mapping of every entity in these types. The additional entities (Entity1…EntityN) in the sub-module all possess a special attribute, which is connected with the primary key of

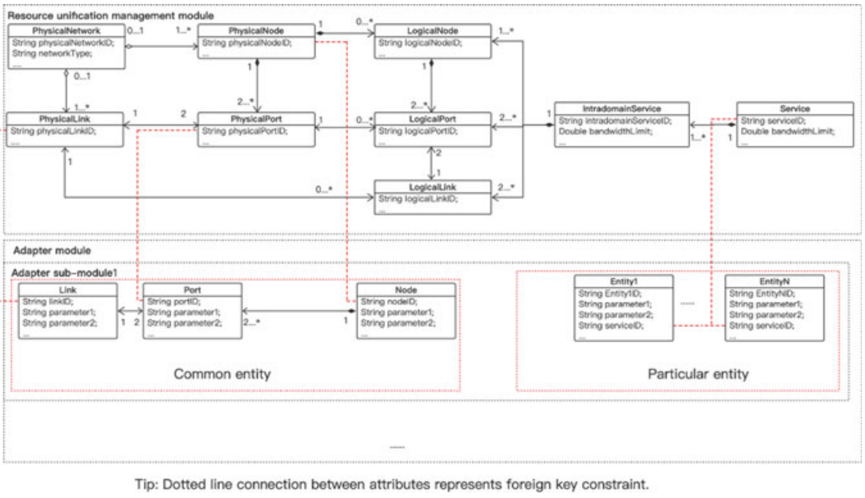

Fig. 83.3 UML class diagram of general controller

a service in the resource unification management module via a foreign key, which indicates their attachments to service.

Among adapter sub-modules, entities of nodes, ports and links belong to common entities of each SDN domain by reason that information of them will be provided however large the internal difference between network devices in different domains. Thus, every sub-module should contain these entities from the perspective of design. These common entities contained in sub-modules possess uncertain extension attributes but with determined relationship of class.

We classify these entities in all SDN domains into common entities and particular entities to summarize the design principle of adapter sub-modules. That is sub-module should include in entities of these four types (links, nodes, ports and additional entities).

83.4.2 Service Implementation Process

Taking creation of a service as an example, the flowchart of implementation is shown in Fig. 83.4. The details and steps are as follows.

1. In order to create a service across multiple domains, other systems or modules require to provide a complete service path and other additional information about this service (such as bandwidth). Through invoking the interface of general controller, service parameters are transmitted into resource unification management module.
2. After receiving request parameters of service, firstly resource unification management module determines whether the syntax and semantic meaning of these parameters are correct. For example, whether service path is correct in current topology or not. After gotten confirmation, parameters including complete service path will be transmitted into adapter manager in adapter module.
3. The adapter manager divides this complete service path into multiple path segments for each SDN domain, and then simultaneously sends parameters containing intra-domain service paths in different SDN domains to all corresponding adapter sub-modules respectively in multi-thread and high-concurrency manner.
4. After one of the adapter sub-modules receives these parameters including intra-domain path in the domain it corresponding to, it will start to performing instruction translation function. By translating, this sub-module will receive invocation sequence of northbound interface of SDN controller in this domain and acquisition procedure of some parameters required when invoking interface. For instance, obtain IDs of head node and tail node in this domain according to the intra-domain service path. And then query their UUIDs for interface invocation by IDs. Besides, it will also receive the process of creating additional entities according to the service information. For example, create a SR tunnel entity

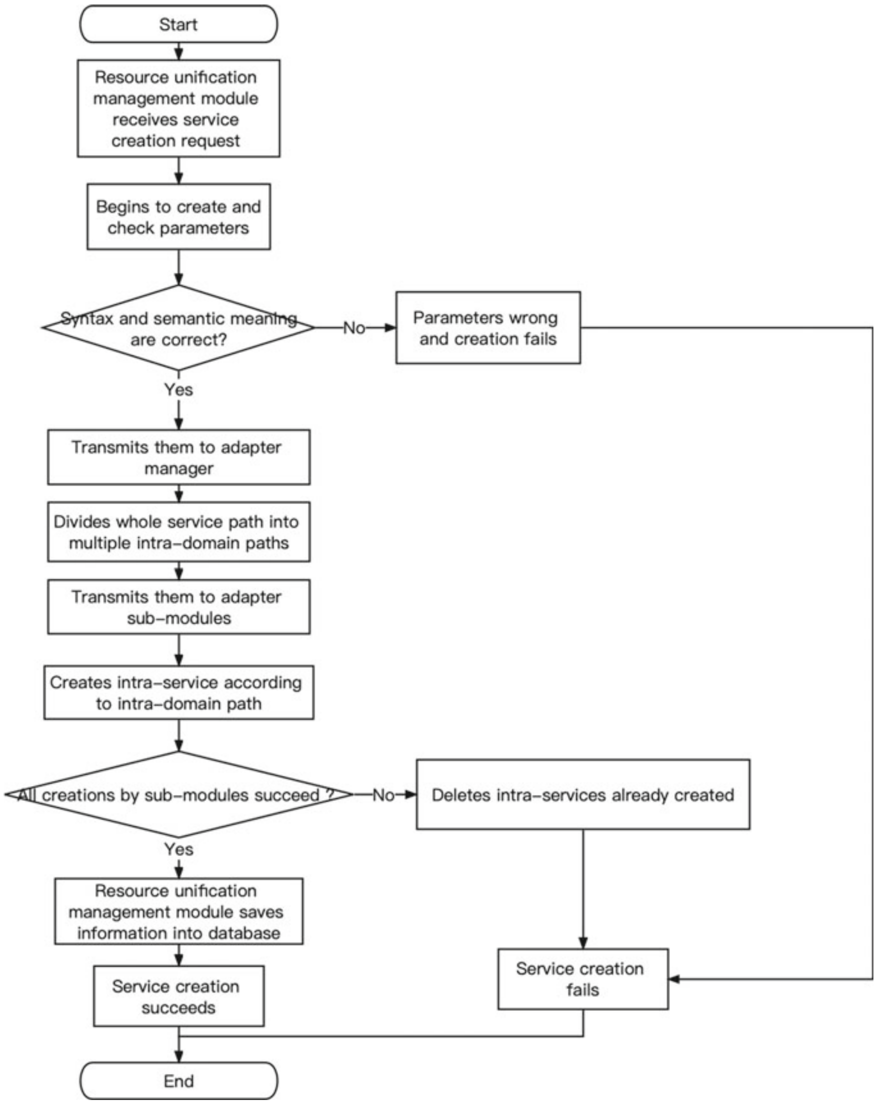
Start
Resource unification management module receives service creation request
Begins to create and check parameters
Syntax and semantic meaning are correct?
No
Parameters wrong and creation fails
Yes
Transmits them to adapter manager
Divides whole service path into multiple intra-domain paths
Transmits them to adapter sub-modules
Creates intra-service according to intra-domain path
All creations by sub-modules succeed ?
No
Deletes intra-services already created
Yes
Resource unification management module saves information into database
Service creation fails
Service creation succeeds
End

Fig. 83.4 Flowchart of service implementation in general controller

according to loopback IPs and UUIDs of these nodes, service bandwidth and other information.

5. After completing the creation of service process in one domain, adapter sub-module waits for the result from SDN controller below. If result indicates success, it will save additional entities, which is created before, associated with the service to database, and send a successful result back to adapter manager. Otherwise,

additional entities information will not be stored, and a failure result will be returned.

6. The adapter manager collects feedbacks of creation from every adapter sub-module. Only when all of them indicate successful creation, does the manager send successful result to resource unification management module, which will start to create corresponding abstract entities, namely logical nodes, logical ports, logical links, intra-domain services and complete service as well as save them to database. Finally successful message will be sent to upper-layer systems or modules.
7. When any adapter sub-module returns a failure, the adapter manager records other adapter sub-modules that return a successful result. And it sends deletion requests to these sub-modules, which will then execute the deletion process after receiving the request. Through instruction translation, the sub-modules invoke a series of methods in northbound interfaces of SDN controllers below to clear service configuration deployed in underlying device in corresponding domains. At the same time, additional entities related to this failure service will be erased in database.
8. Finally, the adapter manager sends a failure result to resource unification management module, which will next transmit it to upper-layer systems or modules.

83.5 Instance Analysis

The following section will take creation of service as an example to introduce service processes of adapter sub-modules corresponding to two SDN implementation technologies, OpenFlow and SR-MPLS, respectively.

83.5.1 Design of Service Process of OpenFlow Adapter Sub-module

Taking creation of a service as an example, the flowchart of service process of OpenFlow adapter sub-module is shown in Fig. 83.5. The details and steps are as follows.

1. After receiving service parameters, the OpenFlow adapter sub-module translates them into a series of instruction sequences via instruction translation function. According to these instructions, firstly this sub-module retrieves information of physical ports, which service path passes through in sequence, as well as physical nodes that they belong to. Then it queries information of parameters required by methods in northbound interface, such as names of bridge in these physical nodes, names of these physical ports in bridge and so on. After that, it activates these

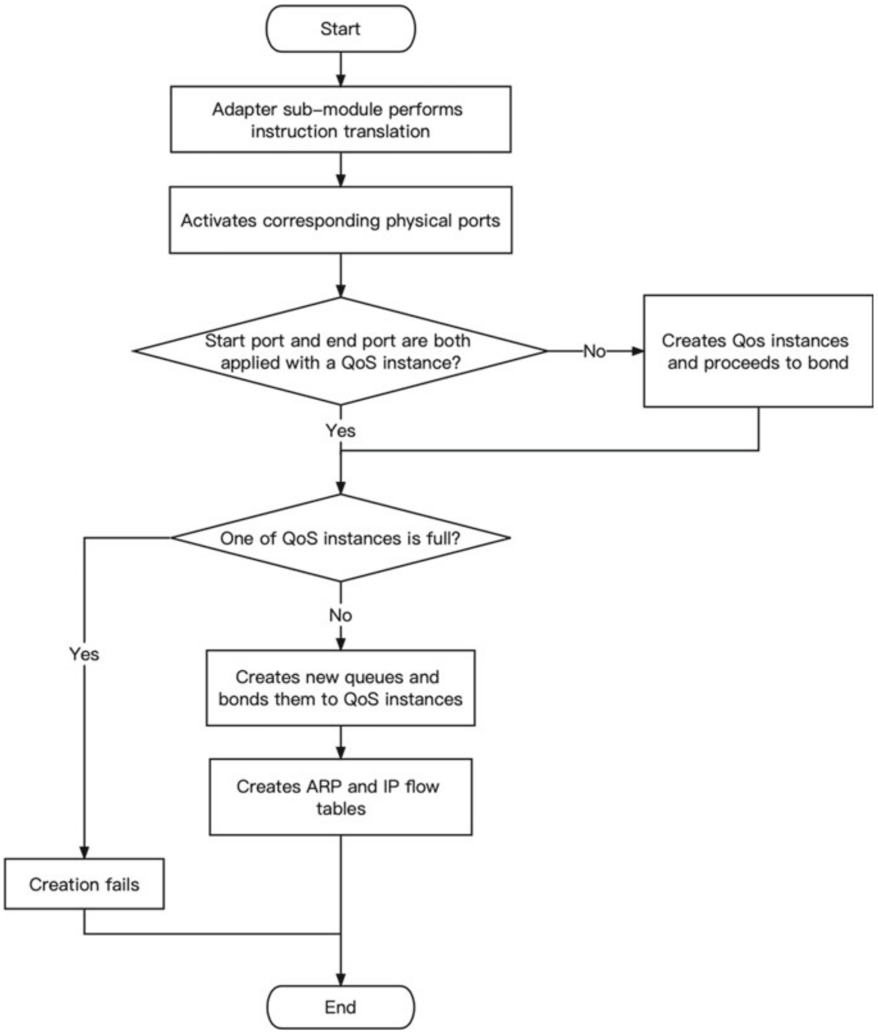
Start
Adapter sub-module performs instruction translation
Activates corresponding physical ports
Start port and end port are both applied with a QoS instance?
No
Creates Qos instances and proceeds to bond
Yes
One of QoS instances is full?
Yes
No
Creates new queues and bonds them to QoS instances
Creates ARP and IP flow tables
Creation fails
End

Fig. 83.5 Flowchart of process of service creation in OpenFlow adapter sub-module

ports in turns by invoking methods in northbound interface of OpenDaylight controller. (No invocation if the port is in open state)

2. This sub-module finds the start port and the end port of service path, and judges whether these two ports are applied with a QoS instance respectively. If at least one port not, the same number of QoS instances will be created by controller interface invocation to be bonded to the ports without QoS instance in underlying devices. Besides, QosID distributed by this sub-module and QosUUID generated by OpenDaylight controller will be saved as extensive attributes of the ports into database. (One port can be only applied with one QoS instance at most.)

3. After confirming that both ports are applied with QoS instances, this module continue to determine whether these QoS instances respectively contain 8 rate-limiting queues (One QoS instance can accommodate 8 rate-limiting queues at most). If one of the QoS instances contains 8 queues, which indicates that at least one port has been fully occupied by services, failure result of service creation will be sent back to the adapter manager. Otherwise, creation process will be proceeded by reason that both ports are able to contain more rate-limiting queue.
4. When continuing to perform creation process, this sub-module creates a rate-limiting queue for these two QoS instances respectively by controller interface invocation. The attributes, such as ID, UUID, queue number (range from 0 to 7), service bandwidth, service ID etc., of these two queues that are regarded as additional entities will be saved to database. And then this sub-module appends queues to related QoS instances in underlying devices by invocation of interface of SDN controller.
5. Finally, this sub-module creates ARP and IP flow tables according to information of ports that service path passes by in sequence. (The parameters required by flow tables include datapath ID of bridge in physical node, OpenFlow ID of ports, IP of source and destination terminals, queue number, etc.)
6. This sub-module succeeds in accomplishing creation process of service, after creating flow tables.

83.5.2 Design of Service Process of SR-MPLS Adapter Sub-module

Taking creation of a service as an example, the flowchart of service process of SR-MPLS adapter sub-module is shown in Fig. 83.6. The details and steps are as follows.

1. After receiving service parameters, the SR-MPLS adapter sub-module translates them into a series of instruction sequences via instruction translation function. And then it determines the head node, the tail node and SR labels of links passed by according to service path. It invokes methods in northbound interface of iMaster controller to create two unidirectional SR tunnels at the head node and the tail node respectively according to service bandwidth and information above. Besides, the information of SR tunnels, which are regarded as additional entities by bonded with service ID, will be saved in database.
2. This sub-module invokes controller interface to generate other resources required by service in underlying devices. These are ACL rule, traffic classification and traffic behavior. Besides, the information of these entities, which are regarded as additional entities by bonded with service ID, will be saved into database. And then it continues to invoke methods in northbound interface of controller, so that not only ACL rule, which IPs of source and destination terminals are appended in, is attached to a traffic classification as a matching condition, but also action

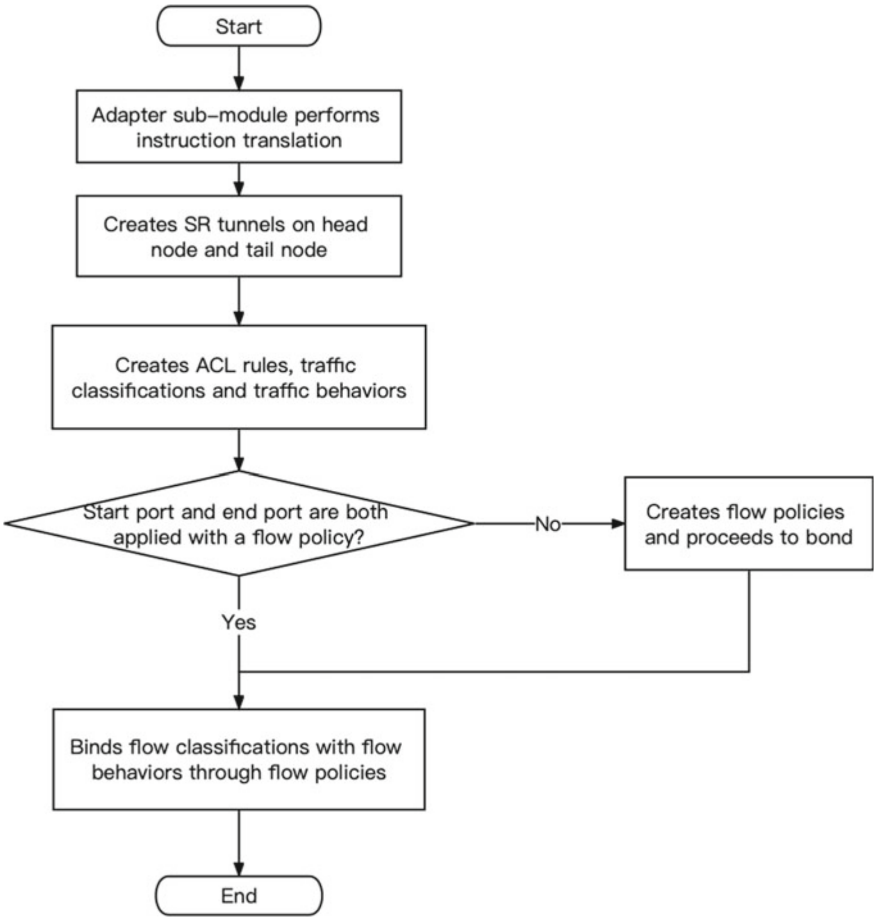
Start
Adapter sub-module performs instruction translation
Creates SR tunnels on head node and tail node
Creates ACL rules, traffic classifications and traffic behaviors
Start port and end port are both applied with a flow policy?
No
Creates flow policies and proceeds to bond
Yes
Binds flow classifications with flow behaviors through flow policies
End

Fig. 83.6 Flowchart of process of service creation in SR-MPLS adapter sub-module

of the flow behavior is set to redirect to a SR tunnel. The process above will be performed twice by reason that there are two SR tunnels.

3. This sub-module finds the start port and the end port of service path, and judges whether these two ports are applied with a flow policy in inbound direction respectively. (One port can be only applied with one flow policy at most). If at least one port not, the same number of flow policies will be created by invocation of controller interface to be bonded to these ports. Besides, flow policy name will be saved as an extensive attribute of the port into database.
4. Finally, this sub-module invokes controller interface, so that flow policies applied on the start node and the end port are used as carriers to bind flow classifications with flow behaviors in underlying devices. After that, when packets of this service enter the start port or the end port, the flow policy bound to this port will come into effect. The packets will be matched by the ACL rule on the flow classification, and

finally be redirected to corresponding SR tunnel. In the end, underlying devices realize communication between terminals of this service.

83.6 Conclusion

Through extracting common traits from different SDN network domains, the general SDN controller system framework proposed in this paper succeeds in unifying and managing network resources to solve the difficulties of orchestration of cross-domain service in multi-domain heterogeneous SDN networks. This system framework divides overall adaptation module into respective sub-modules according to SDN technologies applied in different domains so that discrepancies between technologies only embody in sub-module. Besides, in adapter module, sub-modules provide the same interface to adapter manager, thus with great scalability. When there is a requirement to manage an another SDN domain, it only requires to develop a new sub-module in adapter module according to the same interface and design principle.

However, in the scenario that network devices, which applied with the same SDN implementation technology, locate in multiple SDN domains due to being produced by different manufacturers, it is still necessary to develop multiple sub-modules for corresponding domains because of slight differences between northbound interfaces of SDN controllers from different manufacturers. The further optimization object is to ensure that differences between SDN network domains, in which devices are produced from different manufacturers but with the same SDN implementation technologies, should be only embodied in instruction translation function in adapter sub-module. In this scenario, it is still necessary to develop quite different adapter sub-modules to manage devices applied with different SDN implementation technologies.

References

1. Yushen, W., Kai, Y., Wenyun, X.: Research on cloud network integration implementation scheme. Telecom Power Technol. **38**(1), 6 (2021)
2. Caiming, L., Rong, W.: network analysis of could computing under SDN context. China New Telecommun. **22**(4), 1 (2020)
3. Nan, C., Yongbing, F., Xiaowu, E., et al.: Research on the technology of application—oriented cloud data center network. Telecommun. Sci. **30**(9), 6 (2014)
4. Linze, W., Yongbing, F., Zhilan, H., et al.: Design and implementation on orchestrator for heterogeneous SDN data center solutions. Telecommunications Science **34**(11), 9 (2018)
5. Hongwei, S., Fengzhi, H.: Research on network slicing technology based on SRv6. Electric Technol. Softw. Eng. **16**, 4 (2020)
6. Xiongyan, T., Jichun, M., Chang, C., et al.: Achievements and experiences from SDN transformation of China Unicom 169 network. Mobile Commun. **43**(7), 5 (2019)
7. Ruixue, W., Xuetao, X., Sijun, W.: The network architecture and key technologies of China mobile's data center based on SDN. Mobile Commun. **43**(7), 6 (2019)