# Routing as a Service (RaaS): an open framework for customizing routing services

Chao Bu, Xingwei Wang, Hui Cheng, Min Huang, Keqin Li

***Abstract*—With the emergence of large number and various types of Internet applications, the user requirements over the network communication are becoming more and more complicated and personalized. The Internet users usually have continuously changing communication demands for different types of applications. To cope with this challenge, current Internet Service Providers (ISPs) always purchase and operate more physical network devices. Obviously this solution is unsustainable from the economic viewpoint and it also burdens the network management. This has inspired us to deal with the challenge through the software-based approach. In this paper, we adaptively compose appropriate software-based routing functionsinto customized routing services in order to optimize the users' experience over routing service. However, it remains a big challenge to customize routing services flexibly and programmatically under the global view. Inspired by the ideas of Software Defined Networking (SDN) and Network Function Virtualization (NFV), in this paper we propose Routing as a Service (RaaS), an open framework for customizing routing services. Then, we combine the idea of Dynamic Software Product Line (DSPL) to achieve customization for routing services by the two-layered feature model and the orthogonal variability model proposed in this paper. We also formulate formal definitions for routing service customization and carry out a case study to describe the benefit relationships of the user and ISP. Simulation results show that the proposed RaaS is feasible.**

***Index Terms*—Software defined networking, Network function virtualization, Routing as a service, Dynamic software product line, Feature model, Orthogonal variability model**

## I. Introduction

With the rapid development of Internet technologies and continuously expanding network size, many new types of network applications are emerging. Meanwhile, the user communication requirements for these different types of applications also become more and more complicated and personalized. The current end-to-end communication mode to support these applications is implemented by the cooperation of physical network devices (switches, routers, dedicated servers, etc.) and diversified routing functions (packet scheduling, bandwidth allocation, traffic shaping, transcoding, admission control, etc.) running on them. The users frequently change their communication requirements. To deal with this challenge, ISPs have to keep purchasing and operating new physical network devices, leading to high capital expense (CAPEX) and operating expense (OPEX) for ISPs [1]. Therefore, ISPs need a sustainable method which can satisfy the user high communication demands with low investment and good time-to-value. In this paper, we propose a new method based on the perspective of diversified software-based routing functions. The idea is to reuse these routing functions and adaptively compose the appropriate ones into customized routing services (CRSs) in order to satisfy the user personalized communication requirements for different types of network applications.

To customize diversified and personalized routing services, a mass of network resources and functions are readily available from the deployment of multiple heterogeneous sub-networks [2], production of special network devices, and innovation of diversified routing functions. However, for flexibly composing routing functions and elastically calculating routing services, there are also many challenges such as the private device interfaces from multiple manufacturers, different routing configuration methods, complex protocol conversions between domains, and passiveness of routing maintenance. Therefore, an adaptive routing service customization framework, which can seamlessly allocate routing functions and customize routing services in a unified way, should be created. In this paper, we propose Routing as a Service (RaaS) as an open framework for customizing routing services.

When customizing routing services, the proposed RaaS framework should be able to control and allocate the overall routing functions through the multi-grained approach to achieve flexibility, independence, and programmability under the global view. The basic ideas of Software Defined Networking (SDN) [3] and Network Function Virtualization (NFV) [4] have been introduced into the proposed RaaS. In fact, both SDN and NFV are dynamic networking paradigms which are easy-to-manage, easy-to-develop, and easy-to-evolve. With the introduction of the decoupling architecture of control plane and data plane in SDN, in RaaS the logically centralized control plane can control the routing

This paragraph of the first footnote will contain the date on which you submitted your paper for review. It will also contain support information, including sponsor and financial support acknowledgment. For example, "This work was supported in part by the U.S. Department of Commerce under Grant BS123456".

The next few paragraphs should contain the authors' current affiliations, including current address and e-mail. For example, F. A. Author is with the National Institute of Standards and Technology, Boulder, CO 80305 USA (e-mail: author@ boulder.nist.gov).

S. B. Author, Jr., was with Rice University, Houston, TX 77005 USA. He is now with the Department of Physics, Colorado State University, Fort Collins, CO 80523 USA (e-mail: author@lamar.colostate.edu).

T. C. Author is with the Electrical Engineering Department, University of Colorado, Boulder, CO 80309 USA, on leave from the National Research Institute for Metals, Tsukuba, Japan (e-mail: author@nrim.go.jp).

globally, simplify routing configuration and function embedding through its programmability feature, and further promote innovative customization of routing services. NFV decouples the software-based network functions from the hardware-based network devices, and shields the differences of the underlying infrastructures by leveraging virtualization technology. Thus, potentially diverse routing functions can be modularized and I/O interfaces can be standardized. They will be further controlled, migrated, reused, and improved in an individual way [5]. The RaaS framework is developed based on the decoupling of control plane and data plane in SDN and the decoupling of network functions from physical network devices in NFV. Therefore, it will span a series of different networking environments.

Each user should havespecific and personalized routing services customization. However, when facing very large number of users, it is impossible for an ISP to calculate and customize each routing service for each user independently and individually. The reason is that ISP cannot afford the high service provision cost incurred by the high consumption of computation and memory resources and that ISP lacks the rapid development capability for new unexpected services. Naturally, the users are also not willing to share the high cost. In this paper, we also introduce the idea of Dynamic Software Product Line (DSPL) [6] into the routing service composition. It can produce mass routing services for large-scale users, meanwhile satisfying each user's personalized communication requirements.

DSPL is a software reuse paradigm aiming to guide the organizations to develop software products from a common set of core assets [7] rather than one by one from scratch [8]. DSPL exploits commonality and variability among a family of software products so as to develop reusable and dynamically reconfigurable core assets of this kind. Thus, DSPL makes it achievable to massively produce software products of a certain domain. With this approach, we consider routing services as software products, and diverse routing functions as optional components that can be assembled into products. Since ISP is interested in its own profit maximization [9] and the user is interested in his/her own service experience optimization [10], the routing service customization mechanism should not only satisfy the application requirements but also consider the benefits of both user and ISP jointly [11]. In view of this, we have developed the routing service product lines (RSPLs) to customize diversified routing services for different types of applications, and satisfy the requirements of both the user and the ISP.

The major contributions of this paper are as follows. (1) An open framework, RaaS, is devised for customizing routing services based on SDN and NFV. RaaS provides an effective and personalized way to assemble appropriate routing functions and further optimize the user experience. (2) A mass-customization method is developed by establishing multiple RSPLs based on DSPL. (3) A two-layered feature model for RSPL is proposed. In the model, requirements from the non-functional feature layer guide the functional feature layer in selecting and composing appropriate functions. (4) An orthogonal variability model for RSPL is proposed considering the requirements of both the user and ISP. It achieves traceability and consistency of the variability at different phases of the software development. (5) The formal definitions and descriptions of routing service customization are formulated. This can also serve as the basis for routing service development. (6) Two evaluation models are proposed to demonstrate how to maximize ISP economic benefit while the optimization of user service experience is taken into account.

The rest of the paper is organized as follows. In Section II, we review the related work and compare our work with them. In Section III, we present the system framework of the proposed RaaS. In Section IV, we describe the details of the RSPL, and present the two-layered feature model and the orthogonal variability model. In Section V, we illustrate routing service customization and describe how to optimize the benefit for both the user and ISP. In Section VI, we present the simulation experiments and results. Finally, section VII concludes the paper.

## II. Related Work

There have been a lot of researches on routing based on the idea of SDN. In [12], an adaptive routing for video streaming (ARVS) with QoS support over SDN was proposed. It treats the base layer packets and enhancement layer packets as two levels of QoS flows respectively to reduce packet loss rate and enhance coverage under various network loads. In [13], based on SDN, OpenFlow and Network as a Service (NaaS), a software framework named Network Control Layer (NCL) was proposed. It aims to solve QoS limitations while provisioning end-to-end service. In [14], a SDN/OpenFlow control environment was proposed. It provides bandwidth guarantees for priority flows and outperforms best-effort shortest path routing and Integrated Services (IntServ). In [15], a scalable routing and resource management model for SDN-based large network was proposed. Using load balancing and path resizing methods, it focuses on traffic distribution between pairs of paths for improving resource utilization. In [16], a novel architecture and algorithm was proposed to provide QoS enabled services across multiple domains based on SDN, which provides incorporation opportunities to service providing entities. In [17], an end-to-end real-time QoS communication service based on SDN was proposed. It implements a joint routing and access control system. The above researches configure routing based on the logically centralized controller in thef SDN paradigm, which makes it easy and flexible to allocate resources and provide services under the global view. However, they don't consider to reuse existing routing functions or further compose the appropriate functions into new routing services with different characteristics. In addition, these researches mainly focus on improving QoS without considering the user personalized requirements. In contrast, the proposed RaaS combines the centralized control idea of SDN with the routing function virtualization of NFV to provide a programmable and extensible framework with diverse reusable routing functions. Thus, according to the user personalized requirements, the routing services can be adaptively customized with appropriate routing functions.

There are also some researches on providing services with the idea of function composition based on NFV and SDN. In [18], an open platform for service chain as a service (OpenSCaaS) was proposed. It uses the benefits of NFV and SDN to enforce service-chaining policy. In [19], a combinatorial optimization model was developed to describe resources and components of dynamic function composition , and further a Markov approximation based distributed algorithm was proposed. In [20], based on the network technologies of NFV and SDN, an architecture model for the Next-Generation Service Overlay Network (NGSON) was proposed. It leverages the envisioned service-oriented abstractions and programming capabilities at the network control layer to optimize the delivery process. In [21], a service chain instantiation framework based on NFV and SDN was proposed. It combines the network functions in the cooperative and optimal way. In [22], with an SDN-enabled NFV built on IEEE 802.1x, a flow-based network access control solution was implemented. It establishes services as sets of flow definitions that are authorized as the result of an end user authentication process. Based on the combination of NVF and SDN， the above researches can develop diversified services by composing various network functions to deal with flows on demands. However, they have not explored the idea of mass-customization for routing services when facing large-scale users with different requirements. They also do not consider the benefits of the user and ISP. In contrast, the proposed RaaS combines DSPL with routing function selection and routing service composition, thus achieves the mass-customization. Furthermore, RaaS takes the benefits of the user and ISP into account when customizing routing services.

There are some researches on customizing services by leveraging DSPL. In [23], a novel service-oriented product line (SOPL) was proposed, which combines feature-oriented analysis with a self-management QoS framework. In [24], a software product line based approach for stateful service selection with transaction and QoS support was proposed. It chooses the best services by matching every task of the workflow to the user preferences and constraints. In [25], a consumer-centred approach was proposed. It integrates product line engineering with service-orientation by adapting feature-oriented product-line engineering to service-oriented development. In [26], a dynamic software adaptation for SOPL was proposed, which uses a dynamic feature model for a family of service-oriented architectures. In [27], a self-adaptive solution named ArCMAPE was proposed, which leverages the ideas from Software Product Line Engineering to support fault-tolerant composition services. The above researches leverage the feature modelling method of DSPL to improve the service customization. However, they mainly focused on the user requirements to establish the feature model but ignored other participants such as the ISPs. In addition, they do not consider the routing problems when customizing services. In contrast, we have proposed a novel two-layer feature model by leveraging DSPL for routing service customization, and taken both the user service experience and ISP profit expectation into account.

## III. The Framework Of RaaS

The proposed RaaS is based on the idea of decoupling control plane from data plane in SDN and decoupling routing functions from physical network devices of NFV. Oriented to routing service, the open framework RaaS consists of four layers, i.e., physical layer, resource layer, routing service layer and application layer. According to the requirements of both the user and ISP for different types of network applications, the four layers cooperate with each other to customize routing services. The detailed system framework of RaaS is illustrated in Fig. 1.

The physical layer is in charge of processing packets by the forwarding rules. It consists of a series of network domains that may be built under different environments with different networking modes. In a fine-granularity view, each network domain consists of various, programmable and NFV-enabled network physical equipment (switches, routers, servers, etc.) and wired/wireless links that interconnect them. Therefore, the underlying physical equipment not only contains general-purpose forwarding functions, but also can be enhanced and programmed with special-purpose packet processing functions. This promotes them to support CRS provisions for different types of applications, and satisfy requirements of both the user and ISP.

The resource layer is established to virtualize the underlying network resources to form the virtualized resource pool. The software-based routing functions are decoupled from hardware-based network physical equipment, thus, the required routing functions can be embedded into the programmable modules of the NFV-enable equipment. In resource pool of this layer, the physical equipment is virtualized to be logical nodes, and the physical links are virtualized to be logical links. Meanwhile, various routing functions are modularly designed with standardized interfaces, by which functions can be reused, extended, and combined together. Furthermore, through the resource layer, network resources can be monitored and allocated in a unified way under the global view by the routing service layer.

The routing service layer is established as the control center, and is in charge of mass-customization for routing services. By leveraging the method of DSPL, we devise diversified RSPLs for different types of applications according to their domain knowledge. The RSPL contains multiple variation points that are corresponding to multiple and feasible selection and composition possibilities for routing services. According to the requirements of both the user and ISP, it can support diversified routing services customization for various applications based on the predefined variation points in RSPL. On the basis of the established RSPLs in RSPL set, the appropriate routing functions are selected and allocated to compose CRSs. In this approach, the mass-customization for routing services can be achieved, and each customized service not only optimizes the user service experience, but also maximizes ISP profit expectation.

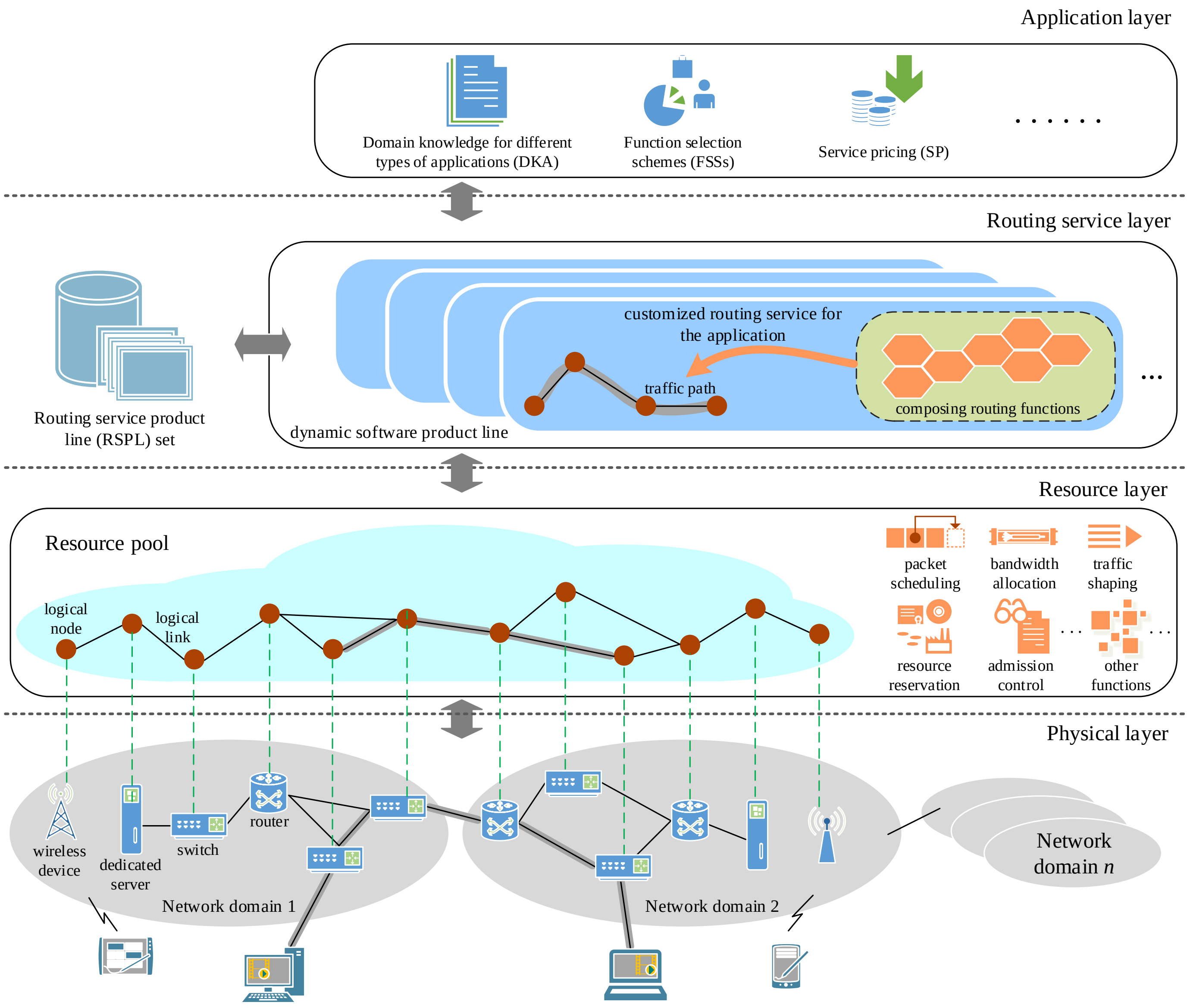


Fig. 1. The framework of the proposed RaaS

The application layer implements routing control logic that supports the routing service layer. In this layer, routing control instances are provided to guide the routing service layer to provide CRSs, for example, three instances are shown in Fig. 1. Domain knowledge of different types of applications is used to establish specific RSPLs. The function selection schemes (FSSs) and the service pricing (SP) strategy are used to provide feasible selection schemes and pricing strategy when customizing routing services with requirements of both the user and ISP considered.

## IV. ROUTING SERVICE PRODUCT LINE

### A. *Variability modelling for RSPL*

When facing various and personalized communication requirements from a large number of users for different types of applications, ISP should not only composes routing services in the form of mass production, but also has to recognize differences among the required services to achieve routing service customization. Thus, RSPL for customizing routing services is proposed, and routing services tailored to individual users' requirements are composed in large-scale. In this approach, based on various routing functions in the resource pool, diversified routing services on the communication paths are composed for different types of applications according to their requirements.

In order to achieve customization for routing services, we predefine commonality and variability of the RSPL. By leveraging the method of DSPL, we establish RSPLs according to DKA and define features in RSPLs according to the externally visible characteristics of routing services. Features can be identified as capabilities, domain technologies, protocols, and algorithms, etc. The commonality among different routing services in a RSPL is modeled as common feature or mandatory feature of RSPL. For example, all applications of VoIP type need the feature of interactivity. The

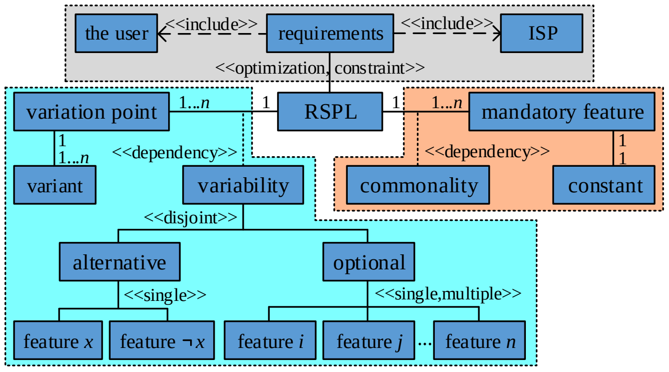


Fig. 2. The relationship among features

variability among different routing services in a RSPL is model as different feature that may be alternative or optional. Alternative features indicate that no more than one feature can be selected for a routing service. For example, connectivity is an alternate feature, because a routing service is either connection-oriented or connectionless. Optional features indicate selectable features for routing services of a certain RSPL. For example, QoS feature is an optional feature, because one routing service can select single or multiple QoS involved algorithms (e.g. packet scheduling, traffic shaping, buffer allocation, etc.) to satisfy different QoS requirements. We define that each of different features (i.e. alternative features or optional features) corresponds to one variation point in RSPL, and the selectable features of each different feature corresponds to the selectable variants of each variation point. In this approach, variation points in RSPL serve as the key to achieve routing service customization according to the personalized requirements. We consider requirements of both the user and ISP to illustrate the relationships among features that are shown in Fig. 2.

In this approach, common features and different features are defined for RSPL. It not only provides reusable basis for customizing routing services, but also makes the customization flexible and extendable. For example, according to selectable variants of each variation point, we consider the corresponding selectable routing functions as variants. Thus, it is easy for RSPL to embed new functions and improve existing functions. Furthermore, maintenance costs can be reduced. For example, once one function is changed, the changes can be delivered to all RSPLs that use the function, which saves extra cost of independently tracing and improving each special related service. In addition, the complexity of routing service management can be simplified. For example, when the user requirement or network status has changed, the routing services do not need to be entirely recomposed from scratch. It only adjusts a few appropriate functions for the changed variation points in RSPL.

## B. *Feature modeling for RSPL*

Feature modeling is the activity of identifying the externally visible characteristics of products in a product line and organizing them into a model, called feature model [28]. In this paper, we consider variability as the key for customizing diversified and personalized routing services, and organize the

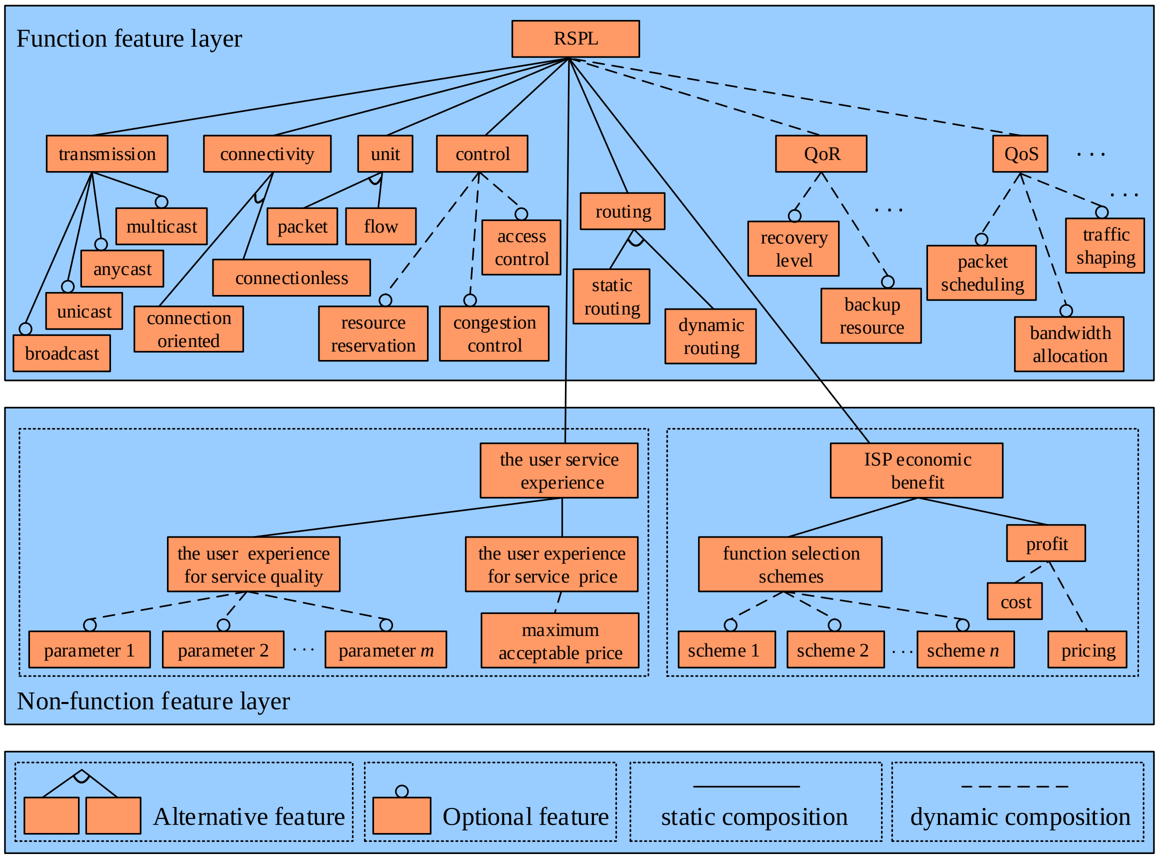


Fig. 3. The feature model of the proposed RSPL

feature model for RSPL based on the different features mentioned above. When composing routing services according to the proposed feature model of RSPL, we should take requirements of both the user and ISP into account, and devise a separate layer for them in the model to guide appropriate function selection and composition. Therefore, we propose a two-layered feature model for RSPL as shown in Fig. 3.

In non-functional feature layer, we consider the user service experience which includes the user experience for service quality (UESQ) and the user experience for service price (UESP). We use service parameters to specify the user requirements for routing services. Thus, UESQ can be calculated by actual parameter values of the routing service. UESP can also be calculated by the relationship between the user maximum acceptable price and the actual service price. Oriented to ISP economic benefit, the cost of CRS will be different according to different function selection schemes, and the actual values of service parameters will also be different due to different function selection. In addition, ISP should price its provided services reasonably to maximize its profit without reducing UESQ. We will later present two simple evaluation models to detail how to choose the appropriate function selection schemes and price the services for ISP in Section 5.

We consider routing services as software products and routing functions as software components to be composed into routing services. The customized characteristics of different routing services composed by a RSPL are achieved based on different features defined in functional feature layer. Thus, according to the requirements from non-functional feature layer, appropriate functions in the virtualized resource pool are selected for each corresponding variation points (Quality of Resilience (QoR), control, QoS, ect) are composed into routing services.

## C. *Orthogonal variability modeling for RSPL*

When composing routing services according to requirements of both the user and ISP, the variability has been defined and depicted in feature model of RSPL, and serves as the key for customization. Meanwhile, feature model of RSPL serves as the customization basis for routing services. Furthermore, we

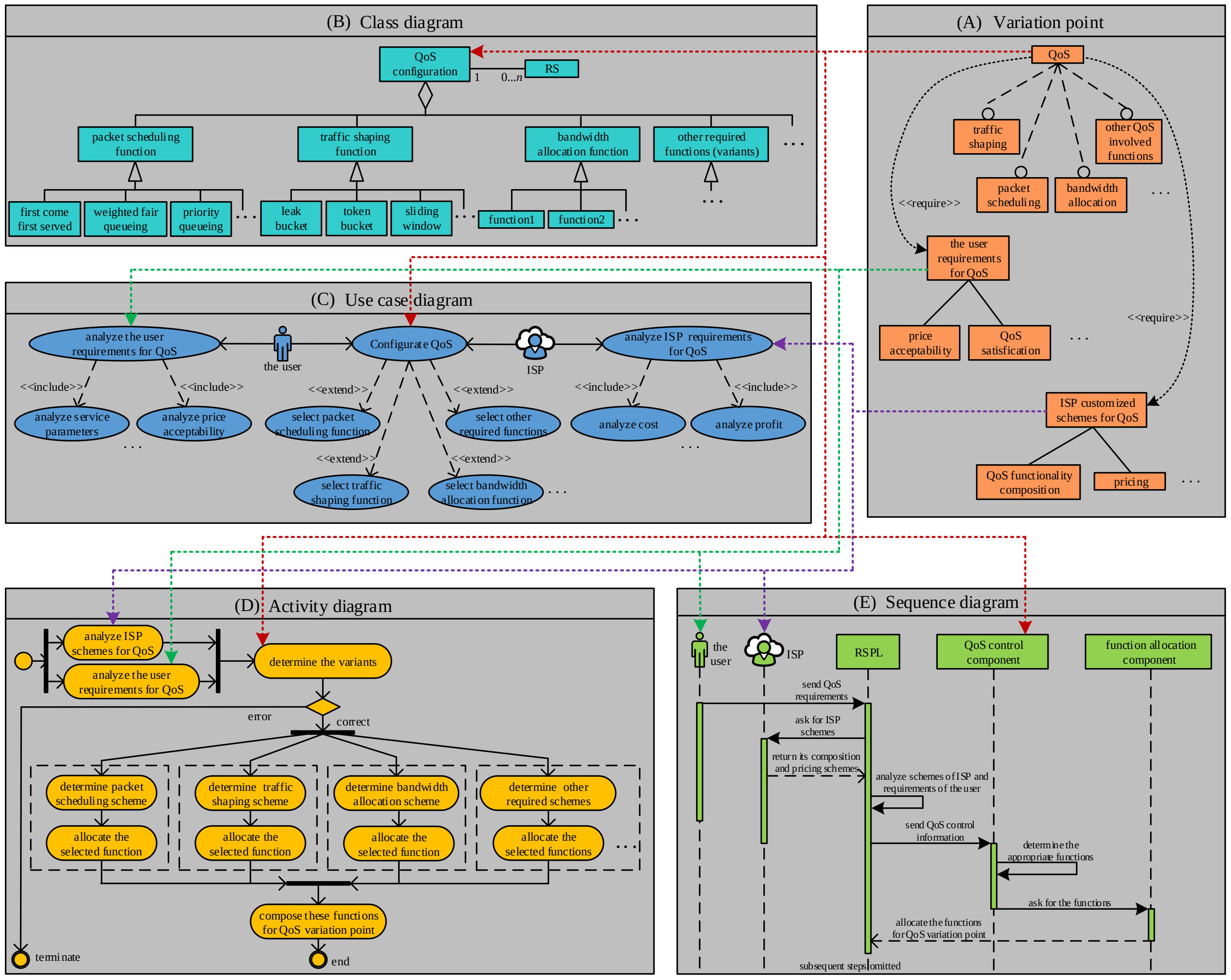

(B) Class diagram
QoS configuration
1
0...n
RS
packet scheduling function
traffic shaping function
bandwidth allocation function
other required functions (variants)
first come first served
weighted fair queueing
priority queueing
leak bucket
token bucket
sliding window
function1
function2
(A) Variation point
QoS
traffic shaping
other QoS involved functions
packet scheduling
bandwidth allocation
<<require>>
the user requirements for QoS
price acceptability
QoS satisfication
<<require>>
ISP customized schemes for QoS
QoS functionality composition
pricing
(C) Use case diagram
analyze the user requirements for QoS
the user
Configurate QoS
ISP
analyze ISP requirements for QoS
<<include>>
<<extend>>
analyze service parameters
analyze price acceptability
select packet scheduling function
select other required functions
analyze cost
analyze profit
select traffic shaping function
select bandwidth allocation function
(D) Activity diagram
analyze ISP schemes for QoS
analyze the user requirements for QoS
determine the variants
error
correct
determine packet scheduling scheme
determine traffic shaping scheme
determine bandwidth allocation scheme
determine other required schemes
allocate the selected function
allocate the selected functions
compose these functions for QoS variation point
terminate
end
(E) Sequence diagram
the user
ISP
RSPL
QoS control component
function allocation component
send QoS requirements
ask for ISP schemes
return its composition and pricing schemes
analyze schemes of ISP and requirements of the user
send QoS control information
determine the appropriate functions
ask for the functions
allocate the functions for QoS variation point
subsequent steps omitted

Fig. 4. The orthogonal variability model for QoS variation point

propose the orthogonal variability model for RSPL to achieve traceability of variability in different phases of software product development models (class diagram, use case diagram, sequence diagram, etc.), and consistency of variability from multiple angles (data, function, action, etc.). In this paper, with requirements of both the user and ISP considered, we take QoS variation point of feature model as an example to establish the orthogonal variability model, which is shown in Fig. 4.

In Fig. 4, based on QoS variation point (Fig. 4(A)) of feature model, we extend variability to other diagrams to trace appropriate routing function selection and composition in this variation point through the proposed orthogonal variability model. In this paper, the involved diagrams in the orthogonal variability model include static view diagrams, such as class diagram (Fig. 4(B)) and use case diagram (Fig. 4(C)), and dynamic view diagrams, such as activity diagram (Fig. 4(D)) and sequence diagram (Fig. 4(E)). These two types of view diagrams (static and dynamic) illustrate variant selection for QoS variation point from structural characteristic and behavioral characteristic respectively. The correspondences of variability among diagrams are also linked with dotted lines as shown in Fig. 4.

In the class diagram of Fig. 4(B), multiple selectable routing functions which can adjust corresponding parameters of QoS, are classified and enumerated. For example, there are many algorithms (first come first served, weighted fair queuing, priority queuing, etc.) which can provide packet scheduling function. One of them is selected by system as one of functional components to be composed into a routing service. In the use case diagram of Fig. 4(C), the use case of analyzing the requirements of both the user and ISP serves as the basis to select appropriate routing functions for QoS variation point. In the activity diagram of Fig. 4(D), it describes the behavior of composing functions and shows the order of activations executed by the involved classes. It also supports multiple thread description when simultaneously allocating multiple functions by RSPL. The interactive process and the exchanged messages among the user, ISP and RSPL in accordance with time sequence are described in the sequence diagram of Fig. 4(E).

## V. Routing Service Customization

When providing routing services, ISP tends to maximize its profit by reducing costs with choosing suitable FSSs and raising prices for customized services with its pricing strategy. However, the chosen FSS and its pricing strategy also influence UESQ and UESP which further affect the user selectivity for the CRS. For example, the user always rejects the service with poor quality and/or high price, so that ISP cannot make any profit. Therefore, it is necessary to maximize ISP profit with the user service experience considered. One way is to formulate and exploit the relationships among the user, ISP, RSPL and CRS. We provide the following formal definitions and descriptions to abstract routing service customization, and they serve as the basis for routing service development. A simple case is used to illustrate how to use them.

### *A. The formal definitions and descriptions*

Definition 1. A user is defined as a four-tuple $\langle u_{id}, crs, R, ep, hp \rangle$. Here, $u_{id}$ is the user's unique identifier; *crs* is the CRS which supports the user communication activity through an application; $R=\{r_1, r_2, ..., r_q | q \in \mathrm{N}_+\}$ is the set of the user requirements on the services, $\mathrm{N}_+$ is the set of positive integers; *ep* and *hp* present the user's estimated price on the service and the user's highest acceptable price to the service respectively. It is used to model the user service experience (USE) for the routing service in the following.

Definition 2. ISP is defined as a four-tuple $\langle DKA, RSPL, FSS, C, sp \rangle$ which provides the routing service to the user. Here, $DKA=\{dka_1, dka_2, ..., dka_p | p \in \mathrm{N}_+\}$ is the set of domain knowledge for different types of applications, each of its elements represents domain knowledge for a certain type of application; $RSPL=\{rspl_1, rspl_2, ..., rspl_p | p \in \mathrm{N}_+\}$ is the set of RSPLs, $rspl_i$ in *RSPL* is the established RSPL according to the corresponding $dka_i$ in *DKA*, and $1 \le i \le p$; $FSS=\{fss_1, fss_2, ..., fss_n | n \in \mathrm{N}_+\}$ is the set of function selection schemes, one element of which is chosen to compose the routing service by ISP; $C=\{c_1, c_2, .., c_n | n \in \mathrm{N}_+\}$ is the set of costs of implementing different schemes in *FSS*; *sp* is the service pricing strategy set by ISP to maximize its profit. It is used to model ISP Economic Benefit (IEB) for providing routing service.

Definition 3. A RSPL is defined as a five-tuple $\langle rspl_{id}, dka_{id}, CF, DF, RF \rangle$. Here, $rspl_{id}$ is the unique identifier of a RSPL; $dka_{id}$ is the corresponding domain knowledge, according to which the $rspl_{id}$ is established, $dka_{id} \in DKA$; $CF=\{cf_1, cf_2, ..., cf_u | u \in \mathrm{N}_+\}$ and $DF=\{df_1, df_2, ..., df_v | v \in \mathrm{N}_+\}$ are common features and different features defined according to $dka_{id}$ for $rspl_{id}$; $RF=\{rf_1, rf_2, .., rf_w | w \in \mathrm{N}_+\}$ is the set of selectable routing functions in $rspl_{id}$ for composing routing services. It is used to model a RSPL which provides the basis for routing service customization.

Definition 4. A CRS is defined as a six-tuple $\langle crs_{id}, Pac, e1, e2, fs, SRF \rangle$. Here, $crs_{id}$ is CRS's unique identifier; $e1$ and $e2$ are the endpoints of the CRS; $Pac=\{pac_1, pac_2, ..., pac_e | e \in \mathrm{N}_+\}$ is the set of packets forwarded by the CRS; *fs* is the forwarding state of *Pac*, including transmission rate, size, loss rate, etc., which has been dealt with by the CRS; *SRF* is the set of the selected routing functions which are composed into the CRS, $SRF \subset RF$. When packets in *Pac* enter the CRS from one endpoint, they are dealt with by the functions in *SRF* and are forwarded to the other endpoint of the CRS under *fs*.

Definition 5. A routing function is defined as a four-tuple $\langle rf_{id}, inp, op, CM \rangle$. Here, $rf_{id}$ is the unique identifier of a routing function, $rf_{id} \in RF$; *inp* and *op* are input port and output port of a function respectively; $CM=\{pullm, pushm, uncertainm\}$ is the set of connection modes of ports. *pullm* represents pull mode and supports a function pulling packets from another function; *pushm* represents push mode and supports a function pushing packets to another function; *uncertainm* represents uncertain mode and serves as the connector between *pullm* and *pushm*. *inp* is either *pullm* or *uncertainm*, and *op* is either *pushm* or *uncertainm*. A *pullm inp* of a function can only connect with an

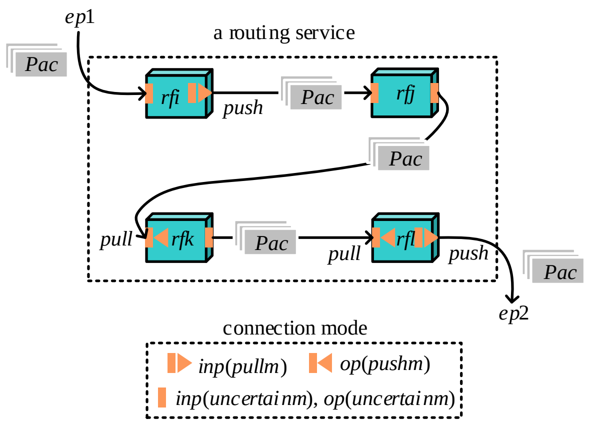


Fig. 5. A routing service

*uncertain op* of another function. A *pushm op* of a function can only connect with an *uncertain inp* of another function.

Definition 6. The operator $\bullet$ is defined to do the operation of selecting one element from a set. For example, $RF \bullet rf_i$ means selecting $rf_i$ from $RF$.

Definition 7. The operator *put* is defined to do the operation of putting $rf_i$ into $crs_j$ by $put(rf_i, crs_k)$.

Definition 8. The operator *del* is defined to do the operation of deleting $rf_i$ from $crs_j$ by $del(rf_i, crs_k)$.

Inference 1. The operation of replacing one function in a service with another function is achieved by doing one $\bullet$operation, one *del* operation and one *put* operation.

Inference 2. The operation of composing a routing service is achieved by doing $\bullet$ operation and *put* operation for $x$ times, here, $x \in N_+$.

Definition 9. The operator *push* is defined to push packets to a function when packets are ready. If a *push* is operated by $rf_i$, $rf_i$ is the source function of the *push*.

Definition 10.The operator *pull* is defined to pull packets from a function when packets are ready. If a *pull* is operated by $rf_i$, $rf_i$ is the destination function of the *push*.

Inference 3. The operation of forwarding packets by a routing service is achieved by doing *pull* operation for $x$ times and *push* operation for $y$ times by the selected functions in the service, here, $x, y \in N_+$.

Proposition 1. The function selection scheme and service pricing strategy for the CRS should maximize IEB with USE optimized.

Proposition 2. A routing service is composed according to a function selection scheme based on a RSPL.

Proposition 3. A routing service can be treated as a directed acyclic graph. The selected functions in the service can be treated as the nodes in the graph and they deal with packets by *push* and *pull* operations. It can be illustrated as Fig. 5.

Proposition 4. An established routing service can be improved by putting, deleting and/or replacing routing functions by its corresponding RSPL without being done from scratch.

Based on the above, the details of RSPL, IEB, CRS and USE are illustrated in Fig. 6.

In the following, we propose the USE evaluation model and the IEB evaluation model to support FSS selection and service pricing strategy determination in order to maximize IEB with USE optimized.

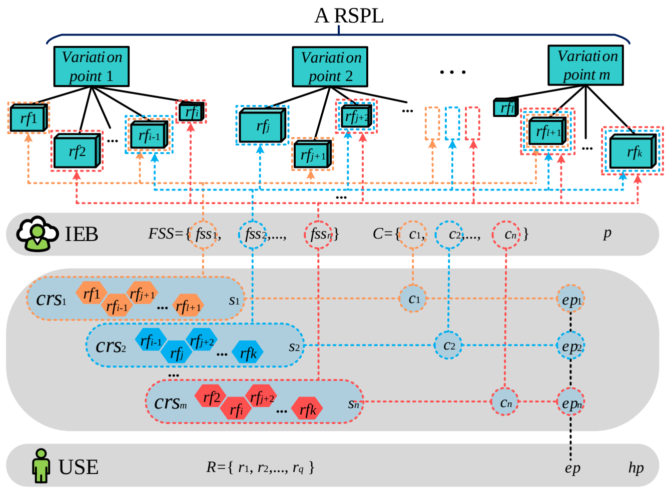


Fig. 6. The illustration of RSPL,

## B. *The USE evaluation model*

The USE consists of UESQ and UESP. The user usually cannot express his requirements accurately and his judgments on things always follow Gaussian distribution [29]. Therefore, in UESQ we use the intervals to represent the inaccurate requirements, that is, $[r_1^l, r_1^h], \ldots, [r_q^l, r_q^h]$ to represent the inaccurate requirements on $r_1, \ldots, r_q$; at the same time we use the Gaussian fuzzy membership function [30] to deal with the users' evaluations on their experience.

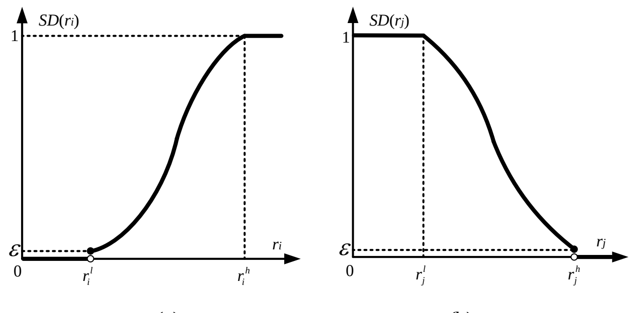
There are two situations on the parameters. The first is that the higher the actual parameter value is, the better the user evaluation to this parameter (e.g. bandwidth) is. The second is that the lower the actual parameter value is, the better the user evaluation to this parameter (e.g. delay) is. Assume that $r_i$ ($1 \le i \le q$) belongs to the first situation, we define the user satisfaction degree for the value of $r_i$ as $SD(r_i)$.

$$SD(r_i) = \begin{cases} 0\,, & r_i < r_i^l \\ \varepsilon\,, & r_i = r_i^l \\ e^{-\frac{(r_i^h - r_i)^2}{(r_i - r_i^l)^2}}\,, & r_i^l < r_i < r_i^h \\ 1\,, & r_i \ge r_i^h \end{cases} \tag{1}$$

Assume that $r_j$ ($1 \le j \le q, j \ne i$) belongs to the second situation, we define the user satisfaction degree for the value of $r_j$ as $SD(r_j)$.

$$SD(r_j) = \begin{cases} 1\,, & r_j \le r_j^l \\ 1 - e^{-\frac{(r_j^h - r_j)^2}{(r_j - r_j^l)^2}}\,, & r_j^l < r_j < r_j^h \\ \varepsilon, & r_j = r_j^h \\ 0\,, & r_j > r_j^h \end{cases} \tag{2}$$

Here, $0 < \varepsilon << 1$. The situations of Eq. (1) and Eq. (2) are



(a) $r_i$ (b) $r_j$

Fig. 7. The user satisfaction degrees for the actual values of $r_i$ and $r_j$

shown in Fig. 7(a) and Fig. 7(b) respectively.

Therefore, the UESQ is defined as follows.

$$UESQ=\sum_{l=1}^{q}\omega_{r_l}\times SD(r_l), \tag{3}$$

Here, $\omega_{r_l}$ indicates the relative importance of $r_l$ to the UESQ, $0\le\omega_{r_l}\le 1, \sum_{l=1}^{q}\omega_{r_l}=1$.

In UESP, we consider it is affected by the actual price $p$ set by ISP, the user estimated price $ep$, and the user highest acceptable price $hp$ for the service. Thus, the UESP is defined as follows.

$$UESP=\begin{cases}1, & p\le ep\\ 1-\dfrac{p-ep}{hp-ep}, & ep<p<hap\\ \varepsilon, & p=hp\\ 0, & p>hp\end{cases} \tag{4}$$

Here, $0<\varepsilon<<1$.

In this approach, UESQ and UESP should be considered together to determine USE, either of them is too low to cause bad service experience to the user. Therefore, we define USE as follows.

$$USE=UESQ\times UESP \tag{5}$$

## C. The IBE evaluation model

In order to maximize IBE, ISP should choose suitable FSS and price the service reasonably. Assume that the cost of composing service $crs_i$ by $fss_i$ is $c_i$, here, $fss_i\in FSS, ci\in C$. And $p$ is the price set by ISP for the service composed by $fss_i$. Thus, ISP expect profit $pro_i$ is defined as follows.

$$pro_i=(p-c_i)\times USE(crs_i,p) \tag{6}$$

Here, $USE(crs_i,p)$ is the USE for the service $crs_i$ composed by $fss_i$ with $p$ according to Eq. (5), and it serves as the probability that the user accepts the service. According to Eq. (3) and Eq. (4), Eq. (6) can be extended as follows.

$$\begin{aligned}pro_i&=(p-c_i)\times\big(UESQ(crs_i)\times UESP(p)\big)\\ &=(p-c_i)\times\begin{cases}UESQ(crs_i), & p\le ep\\ \begin{pmatrix}UESQ(crs_i)\\ \times\left(1-\dfrac{p-ep}{hp-ep}\right)\end{pmatrix}, & ep<p<hp\\ UESQ(crs_i)\times\varepsilon, & p=hp\\ 0, & p>hp\end{cases}\end{aligned} \tag{7}$$

Here, $0<\varepsilon<<1$.

The user knows the actual value of $ep$, however, ISP does not know it. Assume that ISP knows the user $hp$ according to its historical transaction experiences. For ISP, the value of $ep$ belongs to $[0,hp]$ and obeys a certain distribution function $F$. Therefore, ISP uses $F(ep)$ as the user estimated price, and makes its pricing strategy to maximize its profit according to $F(ep)$. Thus, replacing $ep$ in Eq. (7) with $F(ep)$ as follows.

$$\begin{aligned}pro_i'&=(p-c_i)\times\big(UESQ(crs_i)\times UESP(p)\big)\\ &=(p-c_i)\times\begin{cases}UESQ(crs_i), & p\le F(ep)\\ \begin{pmatrix}UESQ(crs_i)\\ \times\left(1-\dfrac{p-F(ep)}{hp-F(ep)}\right)\end{pmatrix}, & F(ep)<p<hp\\ UESQ(crs_i)\times\varepsilon, & p=hp\\ 0, & p>hp\end{cases}\end{aligned} \tag{8}$$

In Eq. (8), $c_i$ is known, and $UESQ(crs_i)$ can be obtained by Eq. (3). When $p=map$ and $pi>map$, $pro_i$ is approaching to 0. Therefore, the maximum value of profit can be obtained under $p\le F(ep)$ or $F(ep)<p<map$.

Under the condition of $p\le F(ep)$, ISP can maximize its profit $pro_i^{\max}$ with $p=F(ep)$.

$$pro_i^{\max}=\big(F(ep)-c_i\big)\times UESQ(crs_i) \tag{9}$$

Under the condition of $F(ep)<p<map$, ISP can maximize its profit $pro_i^{'\max}$ with $p=(map+c_i)/2$ according to the first-order condition.

$$pro_i^{'\max}=\frac{(hp-c_i)^2}{4\big(hp-F(ep)\big)^2}\times UESQ(crs_i) \tag{10}$$

Then, the maximum profit $pro_i^{*\max}$ can be obtained by comparing $pro_i^{\max}$ in Eq. (9) and $pro_i^{'\max}$ in Eq. (10).

$$pro_i^{*\max}=\begin{cases}\big(F(ep)-c_i\big)\times UESQ(crs_i), & F(ep)<\dfrac{hp+c_i}{2}\\ \dfrac{(hp-c_i)^2}{4\big(hp-F(ep)\big)^2}\times UESQ(crs_i), & F(ep)\ge\dfrac{hp+c_i}{2}\end{cases} \tag{11}$$

Meanwhile, according to Eq. (9), Eq. (10) and Eq. (11), the pricing strategy $p^{\max}$ that maximizes ISP profit can be obtained as follows.

$$p^{\max} = \begin{cases} F(ep), & F(ep) < \dfrac{hp + c_i}{2} \\ \dfrac{hp + c_i}{2}, & F(ep) \geq \dfrac{hp + c_i}{2} \end{cases} \tag{12}$$

$p^{\max}$ is obtained for each FSS, and the most suitable $fss_i$ with IBE maximized can be chosen by comparing the profits of FSSs according to Eq. (6).

## VI. The Simulation Experiments

### A. *The simulation setup*

To simulate the proposed RaaS, we use the Floodlight [31] as the control center of the framework for customizing routing services. It generates rules and distributes them to the switches on application communication paths. The rules contains fields of matching and fields of instructions. The former is used to match requests for routing services. The latter contains action instructions (forwarding to controller, dropping, allocating functions, ordering actions, etc.) that are used to deal with packets, and function IDs that identify functions have been embedded.

We use OpenFlowClick [32] to simulate the switch. OpenFlowClick is a kind of software-based switch which is created based on Click modular router [33] with OpenFlowClick element [34] embedded into it. The Click router is extendable, programmable, and assembled by a series of packet processing modules called elements that can be flexibly selected and added. OpenFlowClick element allows a controller to install rules to guide packet processing through multiple selected routing functions that serve as elements in Click router. It also allows multiple Click routers being controlled by one single controller.

The simulation environment is established on Linux platform (Intel core i5 3.3GHz, 16GB DDR3 RAM) with a floodlight controller running on it, and two OpenFlowClick nodes and two hosts running on virtual machines established on the platform as shown in Fig. 8. We take the variation point of QoS as example, and add corresponding routing functions (i.e. algorithms of bandwidth allocation, packet scheduling, queue managing, traffic shaping, and error control) that are created based on Click element model into OpenFlowClick. For comparison purpose, we choose another current popular service composition approach named service-chaining policy to compare with RaaS under the integration of SDN and NFV environment. The chosen approach is OpenSCaaS [18], and we simulate it for composing routing services. Since applications of video streaming type represent 86 percent of global Internet traffic currently [35], we choose this type of applications in this simulation.

### B. *The simulation results*

We compare routing service setup time of the two approaches. We consider setup time is the time overhead from receiving routing service request to providing routing service.

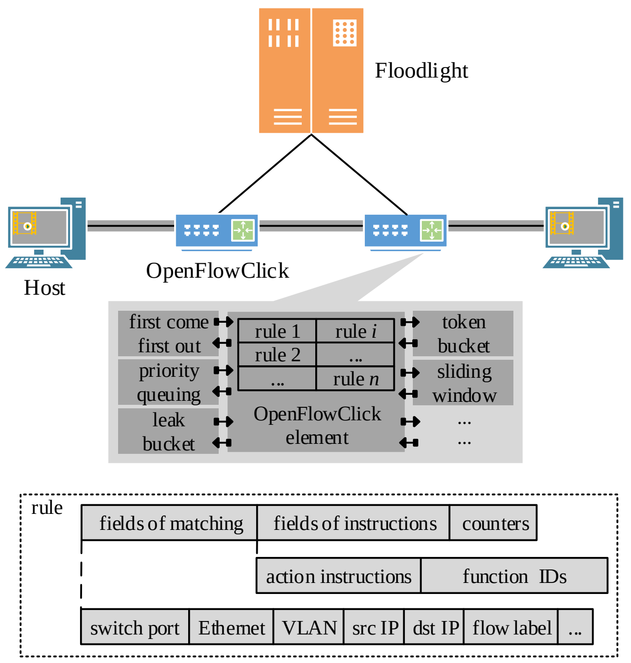
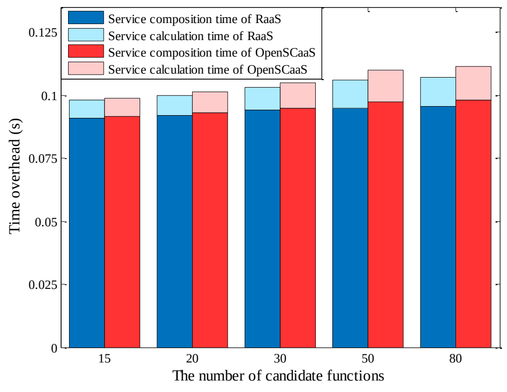
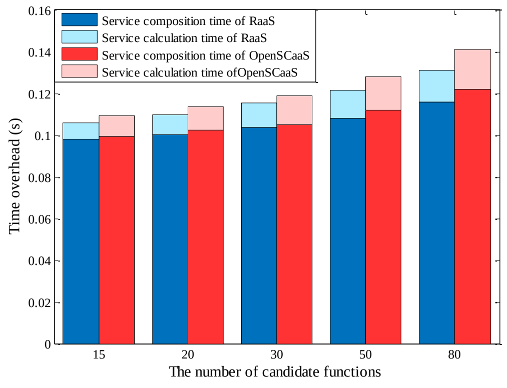


Fig. 8. The simulation environment

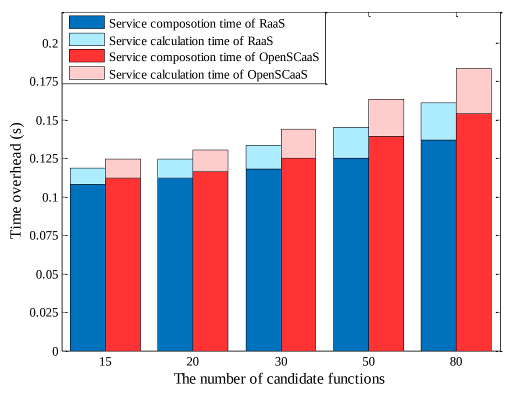
The results are shown in Fig. 9. In Fig. 9, when the number of candidate functions increases, the routing service setup time (the sum of service calculation time and service composition time) increases. When the number of flows supported RaaS and OpenSCaaS increases, the routing service setup time increases too. The time overhead of RaaS always increases slower and



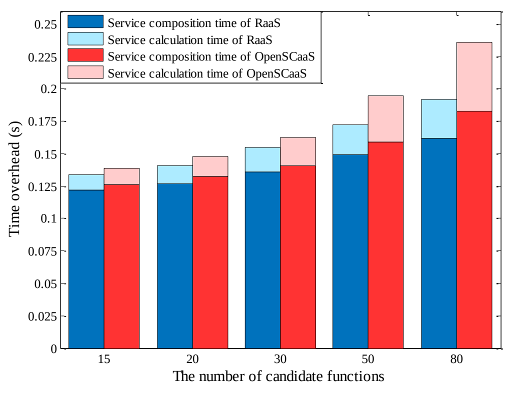
(a) $10^2$ flows



(b) $10^3$ flows

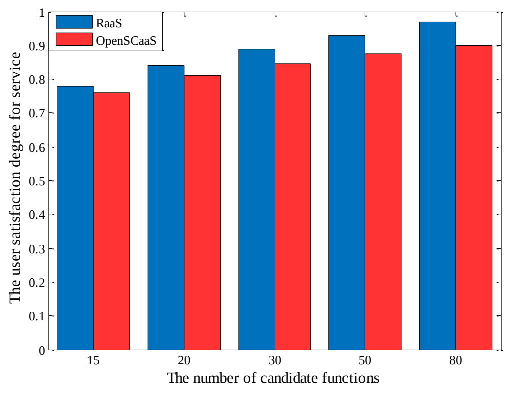


(c) $10^4$ flows

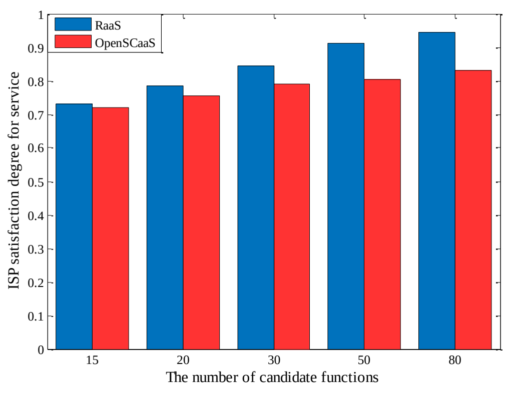


(d) $10^5$ flows

Fig. 9. Time overhead



(a) The user average satisfaction degree

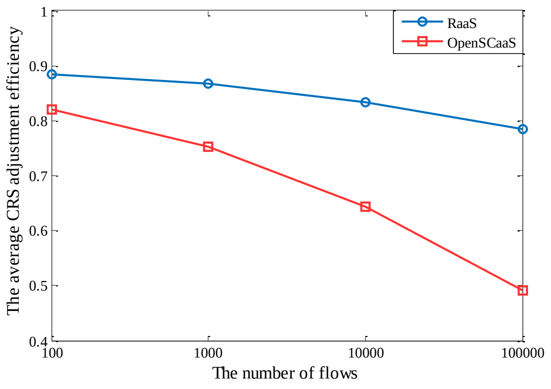


(b) ISP average satisfaction degree

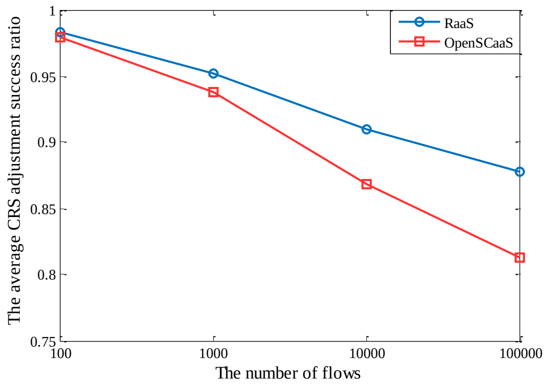
Fig. 10. The average satisfaction degrees for customized services

less than OpenSCaaS, especially when the number of flows increases largely (Fig. 9(c) and Fig. 9(d)). Because based on RSPL, RaaS does not need to calculate and compose all routing services entirely, it just calculates the differences in the variation point and allocates corresponding functions to establish new routing services, which saves much time overhead. However, OpenSCaaS composes each new routing service entirely every time for new flows. In addition, the service calculation time of RaaS is less than 15% of the total service setup time under the peak loads (Fig. 9(d)), while the service calculation time of OpenSCaaS is approximately 25% of the total service setup time at the same situation.

We compare the user and ISP satisfaction degrees for the CRSs composed by the two approaches. The user satisfaction degree is calculate by Eq. (5), and the weights of bandwidth, delay, loss rate and jitter are set as 11.1%, 14.9%, 58.9%, and 15.1% respectively according to [36]. ISP satisfaction degree is the ratio of its actual profit to the service price. The results are shown in Fig. 10. The user and ISP average satisfaction degrees for the services customized by RaaS is much higher than those by OpenSCaaS. The reason is as follows, RaaS customizes routing services in a fine-grained way by leveraging the defined variation points of RSPL to select the classified and appropriate functions independently, which promotes satisfying the user different demands purposely and reducing composition cost. It also take ISP economic benefits into account when selecting functions and pricing services to increase profits. However, OpenSCaaS just considers satisfying the user demands by establishing new entire service chains, which cannot



(a) The average CRS adjustment efficiency



(b) The average CRS adjustment success ratio

Fig. 11. The CRS adjustment efficiency and success ratio

distinguish the fine-grained differences among diversified demands. It also does not consider the issue of optimizing ISP economic benefits when providing services.

The user requirements for CRSs may change when the CRSs are executing. We also compare CRS adjustment efficiency and CRS adjustment success ratio. The CRS adjustment efficiency is 1 minus the ratio CRS adjustment time to CRS setup time. It means the smaller the CRS adjustment time is, the higher the CRS adjustment efficiency is. The CRS adjustment success ratio is the ratio of the number of successfully adjusted CRSs to the total number of CRSs whose requirements are changed. We randomly select 30% of the total CRSs and change the user requirements for them, the results are shown as Fig. 11. In Fig. 11, CRS adjustment efficiencies and success ratios of the two approach decrease when the number of flows increases, and their values of RaaS are always higher than that of OpenSCaaS. The reason is as follows, RaaS just needs to calculate the involved variation points that correspond to the changed requirements based on the RSPL without calculate the entire requirements from scratch. Then, it replaces the involved functions accurately and rapidly according to the certain variation points to adjust CRSs without establishing new CRSs from scratch. However, OpenSCaaS has to consume much more time to calculate entire function configuration according to the changed requirements, and search for suitable functions from candidate functions one by one, which also decreases the CRS adjustment success ratio.

## VII. Conclusion

In this paper, an open framework for customizing routing services based on SDN and NFV named RaaS is proposed. We leverage the idea of DSPL to achieve routing service customization with requirements of both the user and ISP considered. The two-layered feature model for RSPL is proposed and used to compose personalized routing services through diversified routing functions. The orthogonal variability model for RSPL is proposed and used to achieve traceability and consistency of the variability in different phase of the development models. We formulate the formal definitions and descriptions to abstract routing service customization, and illustrate two evaluation models for USE and IBE to optimize benefits of both the user and the ISP.

In order to improve the practicability of our work, we plan to do prototype implementation of the proposed RaaS over the large-scale and real network topologies to further verify its effectiveness and improve its performance. In addition, more effective routing functions selection scheme design and comparison will be investigated in our future research.

## References

[1] Jun Wu, Zhifeng Zhang, Yu Hong, and Yonggang Wen, Cloud radio access network (C-RAN): a primer, IEEE Network, vol. 29, no. 1, pp. 35-41, Jan. 2015.

[2] Ghosh Rumi, and Lerman Kristina, Structure of heterogeneous networks, International Conference on Computational Science and Engineering, vol. 4, pp. 98-105, Vancouver, 29-31 Aug 2009.[never heard this conference, change to a better reference]

[3] Kreutz Diego, M. V. Ramos Fernando, Veríssimo Paulo, Esteve Rothenberg Christian, Azodolmolky Siamak, and Uhlig Steve, Software-defined networking: a comprehensive survey, Proceedings of the IEEE, vol. 103, no. 1, pp. 14-76, Jan. 2015.

[4] Rashid Mijumbi, Joan Serrat, Juan-Luis Gorricho, Niels Bouten, Filip De Turck, and Raouf Boutaba, Network function virtualization: state-of-the-art and research challenges, IEEE Communications Surveys & Tutorials (in publication)

[5] Josep Batalle, Jordi Ferrer Riera, Eduard Escalona, and Joan A. Garcia-Espin, On the implementation of NFV over an OpenFlow infrastructure: routing function virtualization, IEEE SDN for Future Networks and Services (SDN4FNS), pp. 1-6, Trento, 11-13 Nov. 2013.

[6] Nelly Bencomo, Svein Hallsteinsen, and Eduardo Santana de Almeida, A view of the dynamic software product line landscape, Computer, vol. 45, no. 10, Aug. 2012

[7] Svein Hallsteinsen, Mike Hinchey, Sooyong Park, and Klaus Schmid, Dynamic software product lines, Systems and Software Variability Management, pp. 253-260, Mar. 2013.

[8] Kwanwoo Lee, Kyo Chul Kang, and Jaejoon Lee. Concepts and guidelines of feature modeling for product line software engineering. Software Reuse: Methods, Techniques, and Tools, vol. 2319, pp. 62-77, Apr. 2002.

[9] R. T. B. Ma, M. C. Dah, J. C. S. Lui, V. Misra, and D. Rubenstein, On cooperative settlement between content, transit, and eyeball internet service providers, IEEE/ACM Trans. Netw., vol. 19, no. 3, pp. 802-815, Nov. 2010.

[10] C. Tsiaras and B. Stiller, A deterministic QoE formalization of user satisfaction demands (DQX), in 39th IEEE LCN, 2014, pp. 227-235.

[11] T. Ito, M. J. Zhang, V. Robu, S. Fatima, and T. Matsuo, Advances in agent-based complex automated negotiations, in Advances in Agent-Based Complex Automated Negotiations, vol. 233, 2009, pp. 1-20.

[12] Tsung-Feng Yu, Kuochen Wang, and Yi-Huai Hsu, Adaptive routing for video streaming with QoS support over SDN networks, International Conference on Information Networking (ICOIN), pp. 318-323, Cambodia, 12-14 Jan. 2015.

[13] Iris Bueno, José Ignacio Aznar, Eduard Escalona, Jordi Ferrer, and Joan Antoni García-Espín, An OpenNaaS based SDN framework for dynamic QoS control, IEEE SDN for Future Networks and Services (SDN4FNS), pp. 1-7, Trento, 11-13 Nov. 2013.

[14] Slavica Tomovic, Neeli Prasad, and Igor Radusinovic, SDN control framework for QoS provisioning, 22nd Telecommunications Forum Telfor (TELFOR), pp. 111-114, Belgrade, 25-27 Nov. 2014.

[15] M. Rasih Celenlioglu, and H. Ali Mantar, An SDN based intra-domain routing and resource management model, IEEE International Conference on Cloud Engineering (IC2E), pp. 347-352, Tempe, 9-13 March 2015.

[16] Bivas Bhattacharya, and Debabrata Das, SDN based architecture for QoS enabled services across networks with dynamic service level agreement, IEEE International Conference on Advanced Networks and Telecommuncations Systems (ANTS), pp. 1-6, Kattankulathur, 15-18 Dec. 2013.

[17] Jochen W. Guck, and Wolfgang Kellerer, Achieving end-to-end real-time Quality of Service with Software Defined Networking, IEEE 3rd International Conference on Cloud Networking (CloudNet), pp. 70-76, Luxembourg, 8-10 Oct. 2014.

[18] Wanfu Ding, Wen Qi, Jianping Wang, and Biao Chen. OpenSCaaS: an open service chain as a service platform toward the integration of SDN and NFV, IEEE Network, vol. 29, no. 3, pp. 30-35, May. 2015

[19] Peng Wang, Julong Lan, Xiaohui Zhang, Yuxiang Hu, and Shuqiao Chen, Dynamic function composition for network service chain: Model and optimization, Computer Networks, vol. 92, pp. 408-418, Dec. 2015.

[20] Federica Paganelli, Mehmet Ulema, and Barbara Martini, Context-aware service composition and delivery in NGSONs over SDN, IEEE Communications Magazine, vol. 52, no. 8, pp. 97-105, Aug. 2014.

[21] Guozhen Cheng, Hongchang Chen, Hongchao Hu, Zhiming Wang, and Julong Lan, Enabling network function combination via service chain instantiation, Computer Networks, vol. 92, pp. 396-407, Dec. 2015.

[22] Jon Matias, Jokin Garay, Nerea Toledo, Juanjo Unzilla, and Eduardo Jacob, Toward an SDN-enabled NFV architecture, IEEE Communications Magazine, vol. 53, no. 4, pp. 187-193, Apr. 2015

[23] Jaejoon Lee, Gerald Kotonya, and Daniel Robinson, Engineering service-based dynamic software product lines, Computer, vol. 45, no. 10, pp. 49-55, Aug. 2012.

[24] Nadia Gamez, Joyce El Haddad, and Lidia Fuentes, SPL-TQSSS: a software product line approach for stateful service selection, IEEE International Conference on Web Services (ICWS), pp. 73-80, New York, June 27-July 2. 2015.

[25] Gerald Kotonya, Jaejoon Lee, and Daniel Robinson, A consumer-centred approach for service-oriented product line development, Joint Working IEEE/IFIP Conference on Software Architecture, 2009 & European Conference on Software Architecture (WICSA/ECSA 2009), pp. 211-220, Cambridge, 14-17 Sep. 2009.

[26] Hassan Gomaa, and Koji Hashimoto, Dynamic software adaptation for service-oriented product lines, 15th International Software Product Line Conference (SPLC '11), vol. 2, 21-26 Aug. 2011.

[27] Amanda S. Nascimento, Cec´ılia M.F. Rubira, and Fernando Castor, ArCMAPE: a software product line infrastructure to support fault-tolerant composite services, IEEE 15th International Symposium on High-Assurance Systems Engineering (HASE), pp. 41-48, Miami Beach, 9-11 Jan. 2014.

[28] Gerald Kotonya, Jaejoon Lee, and Daniel Robinson, A consumer-centred approach for service-oriented product line development, Joint Working IEEE/IFIP Conference on Software Architecture, 2009 & European Conference on Software Architecture (WICSA/ECSA 2009), pp. 211-220, Cambridge, 14-17 Sep. 2009.

[29] M. C. Platenius, M. Detten, S. Becker, W. Schäfer, and G. Engels, A survey of fuzzy service matching approaches in the context of on-the-fly computing, in Pro. ACM CompArch, 2013, pp. 143-152.

[30] M. H. Hasan, J. Jaafar, and M. F. Hassan, Experimental study on the effective range of FCM's fuzzifier values for web services' QoS data, in ICCOINS 2014, pp. 1-6.

[31] Floodlight [Online]. Available: http://www.projectfloodlight.org/.

[32] OpenFlowClick [Online]. Available: http://archive.openflow.org/wk/index.php/OpenFlowClick.

[33] Eddie Kohler, The Click modular router, ACM Transactions on Computer Systems (TOCS), vol. 18, no. 3, pp. 263-297, Aug, 2000.

[34] Yogesh Mundada, Rob Sherwood, and Nick Feamster, An OpenFlow switch element for Click, Symposium on Click Modular Router, 2009.

[35] H. Yin, Y. Jiang, C. Lin, Y. luo, and Y. Liu, Big data: transforming the design philosophy of future internet, IEEE Network, vol. 28, no. 4, pp. 14-19, July. 2014.

[36] Kim H J, Yun D G. QoE assessment model for video streaming service using QoS parameters in wired-wireless network. Advanced Communication Technology. 2012: 459-464