International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

Y.3015

(04/2016)

SERIES Y: GLOBAL INFORMATION INFRASTRUCTURE, INTERNET PROTOCOL ASPECTS AND NEXT-GENERATION NETWORKS, INTERNET OF THINGS AND SMART CITIES

Future networks

Functional architecture of network virtualization for future networks

Recommendation ITU-T Y.3015

ITU-T Y-SERIES RECOMMENDATIONS

GLOBAL INFORMATION INFRASTRUCTURE, INTERNET PROTOCOL ASPECTS AND NEXT-GENERATION NETWORKS, INTERNET OF THINGS AND SMART CITIES

GLOBAL INFORMATION INFRASTRUCTURE	
General	Y.100–Y.199
Services, applications and middleware	Y.200–Y.299
Network aspects	Y.300–Y.399
Interfaces and protocols	Y.400–Y.499
Numbering, addressing and naming	Y.500–Y.599
Operation, administration and maintenance	Y.600–Y.699
Security	Y.700–Y.799
Performances	Y.800–Y.899
INTERNET PROTOCOL ASPECTS	
General	Y.1000–Y.1099
Services and applications	Y.1100–Y.1199
Architecture, access, network capabilities and resource management	Y.1200–Y.1299
Transport	Y.1300–Y.1399
Interworking	Y.1400–Y.1499
Quality of service and network performance	Y.1500–Y.1599
Signalling	Y.1600–Y.1699
Operation, administration and maintenance	Y.1700–Y.1799
Charging	Y.1800–Y.1899
IPTV over NGN	Y.1900–Y.1999
NEXT GENERATION NETWORKS	
Frameworks and functional architecture models	Y.2000–Y.2099
Quality of Service and performance	Y.2100–Y.2199
Service aspects: Service capabilities and service architecture	Y.2200–Y.2249
Service aspects: Interoperability of services and networks in NGN	Y.2250–Y.2299
Enhancements to NGN	Y.2300–Y.2399
Network management	Y.2400–Y.2499
Network control architectures and protocols	Y.2500–Y.2599
Packet-based Networks	Y.2600–Y.2699
Security	Y.2700–Y.2799
Generalized mobility	Y.2800–Y.2899
Carrier grade open environment	Y.2900–Y.2999
FUTURE NETWORKS	**Y.3000–Y.3499**
CLOUD COMPUTING	Y.3500–Y.3999
INTERNET OF THINGS AND SMART CITIES AND COMMUNITIES	
General	Y.4000–Y.4049
Definitions and terminologies	Y.4050–Y.4099
Requirements and use cases	Y.4100–Y.4249
Infrastructure, connectivity and networks	Y.4250–Y.4399
Frameworks, architectures and protocols	Y.4400–Y.4549
Services, applications, computation and data processing	Y.4550–Y.4699
Management, control and performance	Y.4700–Y.4799
Identification and security	Y.4800–Y.4899

For further details, please refer to the list of ITU-T Recommendations.

Recommendation ITU-T Y.3015

Functional architecture of network virtualization for future networks

Summary

Recommendation ITU-T Y.3015 describes the functional architecture of network virtualization for future networks, covering the specification of user roles, resources and LINPs, functions and their mutual relations, and reference points. An implementation example is also provided.

History

Edition	Recommendation	Approval	Study Group	Unique ID*
1.0	ITU-T Y.3015	2016-04-06	13	11.1002/1000/12712

Keywords

Functional architecture, LINP, network virtualization.

* To access the Recommendation, type the URL http://handle.itu.int/ in the address field of your web browser, followed by the Recommendation's unique ID. For example, http://handle.itu.int/11.1002/1000/11830-en.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure, e.g., interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had not received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database at http://www.itu.int/ITU-T/ipr/.

Table of Contents

Recommendation ITU-T Y.3015

Functional architecture of network virtualization for future networks

1 Scope

This Recommendation describes the functional architecture of network virtualization, which enables management and control of logically isolated network partitions (LINPs) over shared physical networks and supports programmability of services on LINPs. It covers the specification of user roles, resources and LINPs, functions and their mutual relations, and reference points.

2 References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

[ITU-T Y.3001] Recommendation ITU-T Y.3001 (2011), *Future networks: Objectives and design goals.*

[ITU-T Y.3011] Recommendation ITU-T Y.3011 (2012), *Framework of network virtualization for future networks.*

[ITU-T Y.3012] Recommendation ITU-T Y.3012 (2014), *Requirements of network virtualization for future networks.*

3 Definitions

3.1 Terms defined elsewhere

This Recommendation uses the following terms defined elsewhere:

3.1.1 LINP operator [ITU-T Y.3012]: A network operator that creates, programs, configures, manages, and terminates network services on a given LINP.

3.1.2 logically isolated network partition (LINP) [ITU-T Y.3011]: A network that is composed of multiple virtual resources which is isolated from other LINPs.

3.1.3 network virtualization [ITU-T Y.3011]: A technology that enables the creation of logically isolated network partitions over shared physical networks so that heterogeneous collection of multiple virtual networks can simultaneously coexist over the shared networks. This includes the aggregation of multiple resources in a provider and appearing as a single resource.

3.2 Terms defined in this Recommendation

None.

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

API Application Programming Interface

ASIC Application-Specific Integrated Circuit

FPGA	Field-Programmable Gate Array
GRE	Generic Routing Encapsulation
ID	Identifier
IP	Internet Protocol
LINP	Logically Isolated Network Partition
LMI	LINP Management Interface
NFV	Network Functions Virtualization
NNI	Network-to-Network Interface
OS	Operating System
PRI	Programmer-to-Redirector Interface
QoS	Quality of Service
SDN	Software-Defined Networking
SMI	Service Management Interface
UNI	User-to-Network Interface
VLAN	Virtual Local Area Network
VM	Virtual Machine
VMI	Virtual resource Management Interface
VNF	Virtualized Network Function
XML-RPC	Extensible Markup Language-Remote Procedure Call

5 Conventions

None.

6 Overview of functional architecture

This clause provides an overview of the functional architecture of network virtualization. Figure 1 shows the components of the architecture classified into the following four categories: user roles, resources, functions, and reference points.

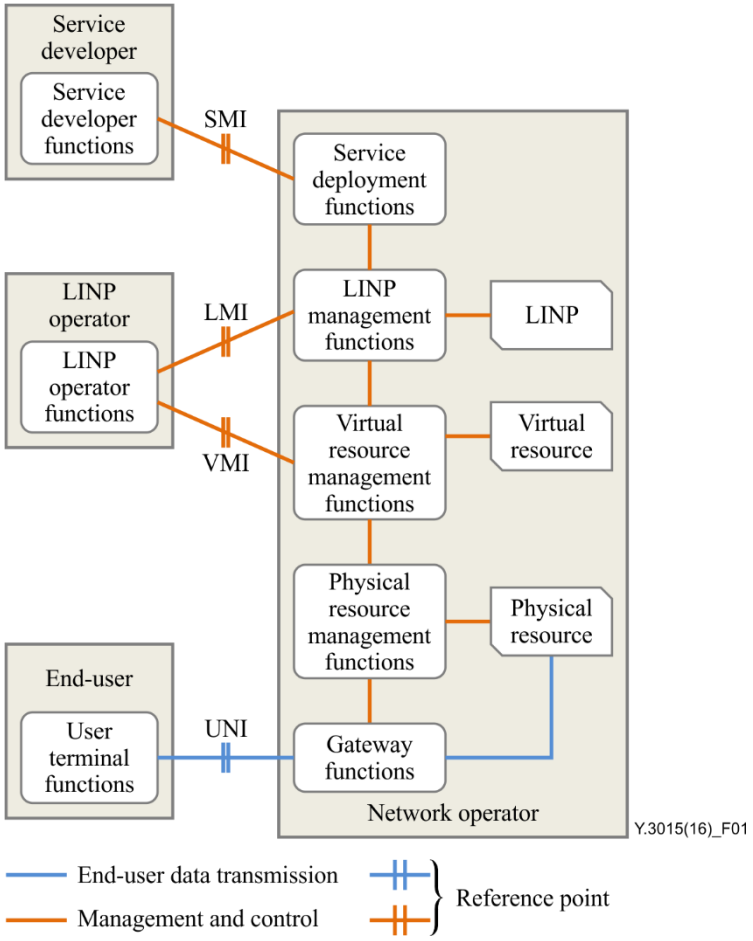

Figure 1 – User roles, resources, functions, and reference points in the functional architecture of network virtualization

A resource is either a physical or a virtual component of a virtual network that executes specific functionality and is represented with specific attributes. Three types of resources are introduced in the functional architectures: physical resource, virtual resource, and LINP. All the three types of resources are managed with corresponding management functions, which are usually implemented in management systems.

NOTE 1 – In Figure 1, end-user data transmission is represented with the blue lines and management and control with the orange lines, as shown in the legend of symbols. Note that the blue lines are applied only to physical resources and functions that manage them. This way of representation makes it clear for which purpose each reference point is specified. For example, the user-to-network interface (UNI) reference point is specified for end-user data transmission, whereas the virtual resource management interface (VMI) is specified for management and control. The same representation is applied throughout this document. See clause 18 for more information about the reference points.

User roles are classified into one of the following four types: end-user, network operator, LINP operator, and service developer. When LINP federation is performed, another type of role, LINP exchanger, may be involved. These user roles are distinguished logically. Different types of user roles could be played by the same party.

Each user role executes certain functions for the purpose of fulfilling its own responsibilities. When functions of one user role need to access those of another, the accesses are done via reference points.

NOTE 2 – [b-ITU-T Y.3502] describes the reference architecture of cloud computing using two different views, i.e. the user view and the functional view. In [b-ITU-T Y.3502], parties and roles appear in the user view and functions in the functional view. The functional architecture presented in the present Recommendation can be understood as a combination of the two views in the context of [b-ITU-T Y.3502].

As such, party as used in the present Recommendation is similar to party as used in [b-ITU-T Y.3502] and user roles in the present Recommendation can be regarded as a shortcut to roles in the user view in the context of [b-ITU-T Y.3502].

NOTE 3 – In this Recommendation, LINP operators are treated independently from network operators, while [ITU-T Y.3012] describes LINP operators as being network operators. This approach allows for describing individual functions in reference to user roles. In line with [ITU-T Y.3012], the party acting as a network operator can also act as an LINP operator and as a service developer.

6.1 User roles

6.1.1 Network operator

Network operators are responsible for managing physical resources and their abstracted sets, or virtual resources. For this purpose, they implement physical resource management functions and virtual resource management functions, respectively, in their individual administrative domains.

A LINP is built from virtual resources in an administrative domain of a network operator. Accordingly, implementing LINP management functions in an administrative domain is a responsibility of the corresponding network operator. LINP management functions use virtual resource management functions of the same administrative domain to allocate appropriate virtual resources to a specific LINP. Likewise, virtual resource management functions use physical resource management functions of the same administrative domain to allocate appropriate physical resources to a specific virtual resource.

Gateway functions are also implemented by network operators. Gateway functions are used to control accesses from end-users to specific LINPs.

6.1.2 LINP operator

LINP operators activate or deactivate services on LINPs provided by network operators. ALINP operator can also have access to LINP management services provided by a network operator. For this purpose, the LINP operator interacts with virtual resource management functions and LINP management functions of the administrative domain through the virtual resource management interface (VMI) and the LINP management interface (LMI), respectively.

When LINP federation is required, LINP operator functions may use two or more virtual resource management functions, each of which resides in different administrative domains. Examples are presented in Figure 2 and Figure 3.

LINP operator may identify suitable administrative domains and select a specific one, when two or more administrative domains can be used to provision a LINP. In such a case, it is the LINP operator's responsibility to discover and select suitable virtual resources from multiple administrative domains.

6.1.3 Service developer

Service developers design, develop, test and maintain services by using service developer functions. Developed services are deployed on a LINP, or else, on two or more LINPs when LINP federation is performed (as in Figure 2 and Figure 3). Examples of services include data forwarding, data routing, or other kinds of data processing like media transcoding.

Service deployment functions are used to deploy the services. The functions are implemented by network operators (as shown in Figure 1) or by the service developer that programs the service. In the former case, the service management interface (SMI) is provided by network operators. A figure describing the latter case is presented in clause 13 (see Figure 9).

6.1.4 End-user

Once a service is deployed on a LINP, an end-user that has authority to access the LINP becomes ready to utilize it.

User terminal functions deliver an end-user's request to gateway functions of a network operator in order to get authorization to access a LINP. Once being authorized, user terminal functions can send and receive end-user data through the UNI reference point.

6.1.5 LINP exchanger

Two or more LINPs belonging to different administrative domains may be federated so that a service can be provisioned across multiple administrative domains. A LINP exchanger provides LINP federation functions to complete LINP federation [b-VNode Federation].

Figure 2 shows an example of LINP federation where a LINP exchanger federates two LINPs from two different administrative domains. In this example, the LINP operator providing a service on a LINP that belongs to one of the two administrative domains may not have direct access to the LINP management functions residing in the other administrative domain. Federation functions of the LINP exchanger intermediate the process in which the two LINPs and the constituent virtual resources are coordinated to ensure defined connectivity, thus eliminating the need for the LINP operator to have cross-administrative domain access to the LINP management functions.

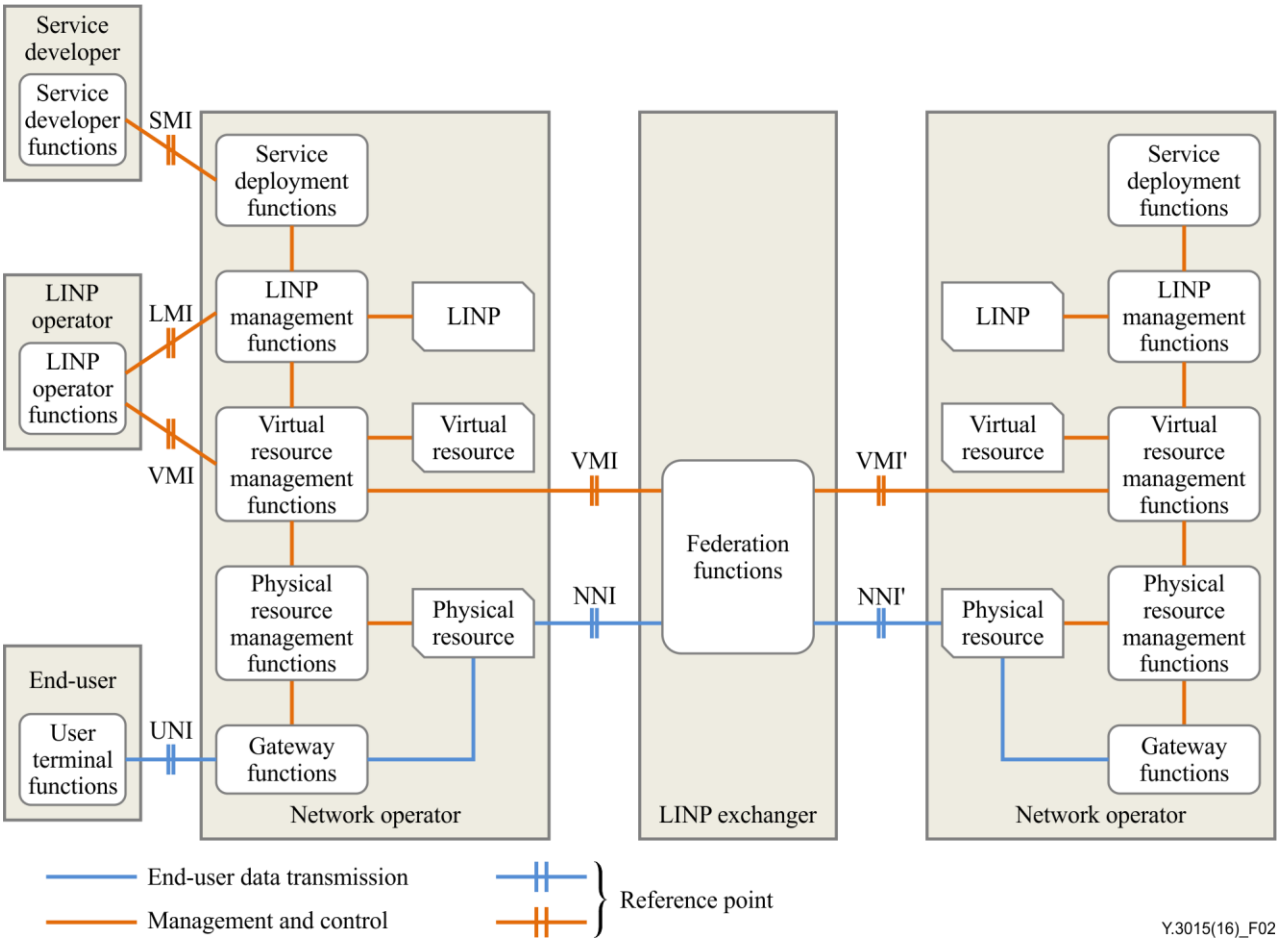

Figure 2 – An example of LINP federation with a LINP exchanger's federation functions

A network-to-network interface (NNI) reference point is specified on each side of the LINP exchanger. Federation functions are recommended to allow the use of different independent protocols at the NNI and VMI reference points between the LINP exchanger and each administrative domain. This is illustrated in Figure 2 where NNI' and VMI' notations are used to represent that the protocol used at the NNI (or VMI) reference point could be different from that used in the NNI' (or VMI') reference point.

6.2 LINP federation without LINP exchangers

LINP federation may be completed without requiring help from the LINP exchanger and its federation functions [b-GENI Architecture]. Figure 3 shows an example of LINP federation where two LINPs from two different administrative domains are federated with no LINP exchanger involved. In this example, an LINP operator uses two LINP management functions that individually reside in the two different administrative domains. The two LINPs and the constituent virtual resources are coordinated to ensure the connectivity. Once the LINP federation is established, a NNI reference point and a VMI reference point are specified between the administrative domains.

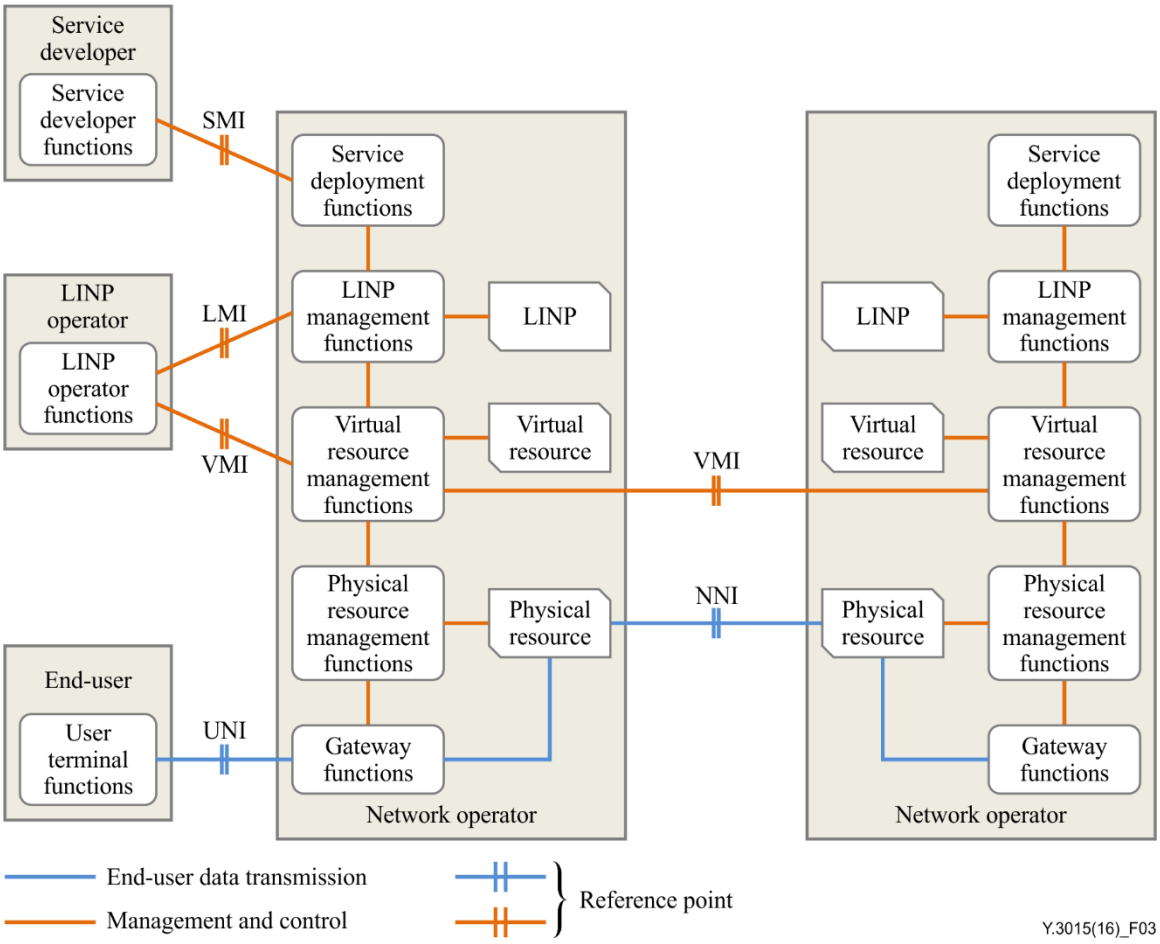

Figure 3 – An example of LINP federation without LINP exchangers

Service deployment in each of the administrative domains is conducted on the federated LINPs by a same service developer. When service deployment functions are implemented by network operators, as in Figure 3, the service developer functions use multiple service deployment functions that individually reside in different administrative domains.

NOTE – [b-ITU-T Y.3511] presents the following three patterns as inter-cloud connections: "inter-cloud peering", "inter-cloud federation", and "inter-cloud intermediary". These patterns are distinguished primarily by whether common application programming interfaces (APIs) are used or not and whether there is an intermediary cloud service provider or not. In this document, the term "federation" is used regardless of whether or not common NNIs and/or VMIs are used or LINP exchangers exist. LINP federation can involve reconciliation of different protocols at the reference points and/or of different administrative policies in any cases. The example shown in Figure 3 can be viewed as equivalent to the inter-cloud peering or the inter-cloud federation as described in [b-ITU-T Y.3511].

7 Resources and LINPs

7.1 Physical resources

As shown in Figure 4, physical resources are classified into four categories: physical link resources, transport resources, computing resources, and storage resources. A physical link resource may consist of a single layer, such as VLAN, or a combination of multiple layers, such as generic routing encapsulation (GRE) and IP. Examples of transport resources include routers and switches. Examples of computing resources include central processing units (CPUs) and memories. Examples of storage resources include devices holding high volume of data.

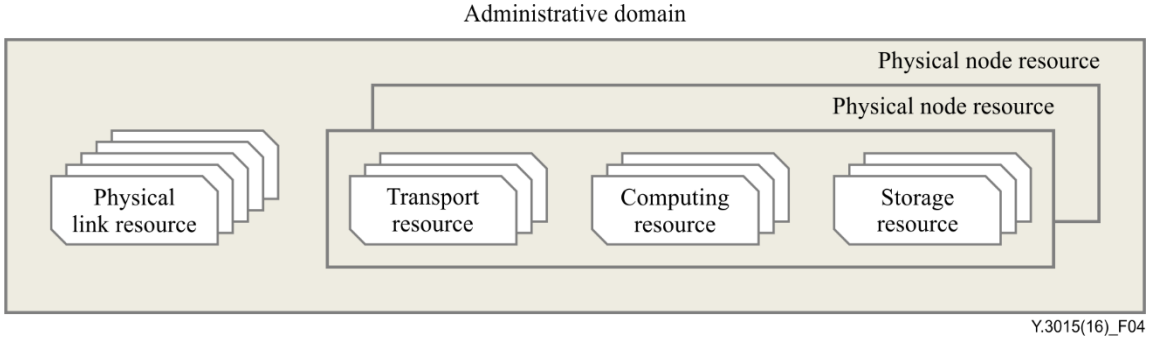

Figure 4 – Categories of physical resources

A physical node resource is composed of transport resources, computing resources, and storage resources in an administrative domain. Data frames/packets are sent and received in a physical node resource through one or more physical link resources.

7.2 Virtual resources

Virtual resources are classified into two categories: virtual link resources and virtual node resources. Virtual link resources are created using physical link resources, and virtual node resources using physical node resources. A virtual link resource may consist of a single layer, such as VLAN, or a combination of multiple layers, such as GRE and IP.

7.3 LINPs

A LINP is created using virtual link resources and virtual node resources. Figure 5 shows an example of a LINP, which consists of three virtual node resources and two virtual link resources.

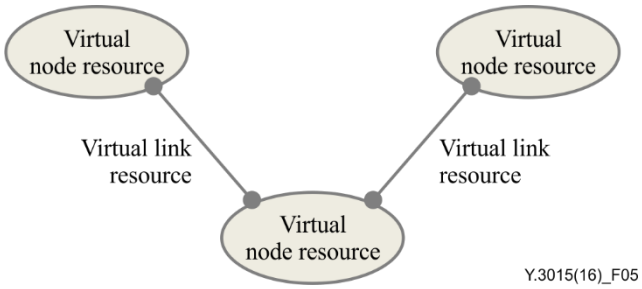

Figure 5 – An example of an LINP

7.4 Allocation and binding

The relationship between a virtual resource and its constituent physical resources is called mapping. Based on the mapping, one or more specific physical resources are allocated to one or more virtual resources. Likewise, the relationship between a LINP and its constituent virtual link and node

resources is called mapping. Based on the mapping, specific virtual resources are allocated to a LINP. Figure 6 depicts these relationships.

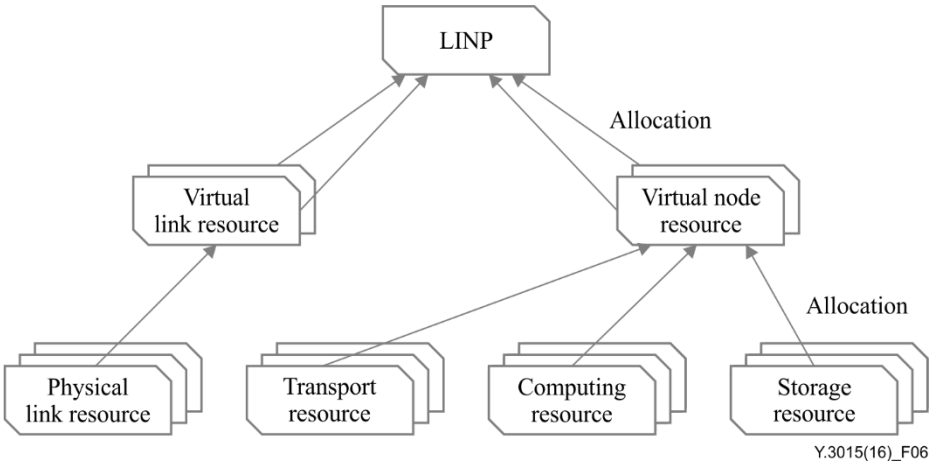

Figure 6 – Allocation

In order for a LINP to become operable, virtual link resources and virtual node resources that are allocated to the LINP have to be interconnected logically. The interconnection is completed by binding the virtual link resources and the virtual node resources together, as depicted in Figure 7.

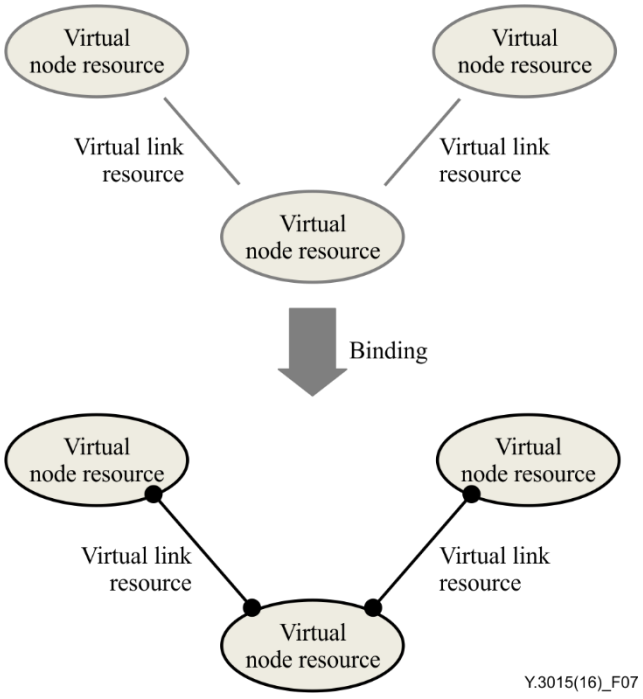

Figure 7 – Binding

8 Physical node architecture

One of the most significant challenges in the node architecture that instantiate LINPs out of physical resources is a scalable design in terms of the number of LINPs to be instantiated.

A physical node resource has a collection of physical resources, i.e., computing, storage and transport resources. The evolutions of technologies for creating these resources tend to have different speed. For example, transport technologies often make progress with a different speed compared to that of

computation and storage. Also, computing and storage resources are often managed together in programming data processing functionalities.

To construct a scalable node design, a physical node resource is highly recommended to be divided into two hardware components in terms of management of resources: a programmer that manages computing and storage resources and a redirector that manages transport resources, as shown in Figure 8 [b-VNode Whitepaper].

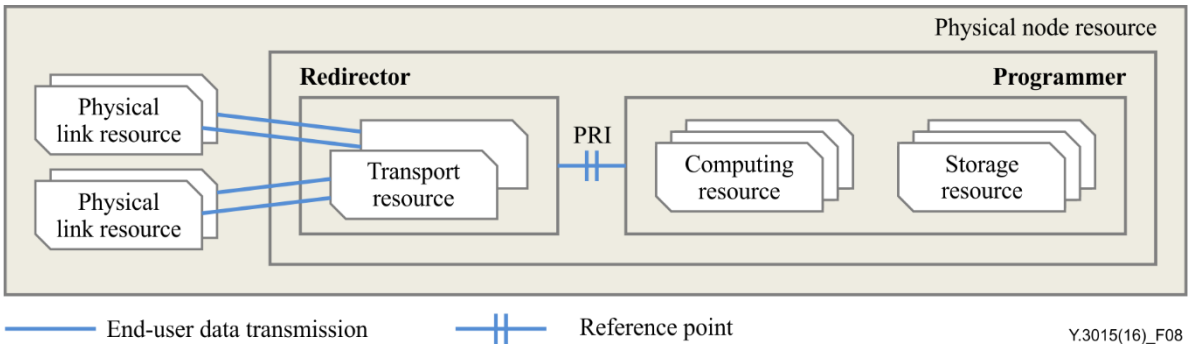

Figure 8 – Physical node architecture

The programmer consists of various kinds of processing components that can be isolated and virtualized to provide an isolate execution environment for data processing and its management. Implementation technologies span across a spectrum of components such as general-purpose servers, network processors, field-programmable gate arrays (FPGAs), and reconfigurable application-specific integrated circuits (ASICs). Each execution environment deploys various software functions to perform data processing.

The redirector consists of transport resources. Virtual link resources with various quality of service (QoS) characteristics and capabilities, such as bandwidth and policing and shaping capabilities with buffering, are created. The redirector may also include programmability to realize virtual, logical links such as tunneling and virtual LAN.

A programmer and a redirector are interconnected through the programmer-to-redirector interface (PRI) reference point which is used to exchange control messages for resource allocation and end-user data exchange.

A physical node design consisting of a programmer and a redirector and the PRI reference point achieves both flexibility and scalability. This node architecture supports independent evolutions of the transport, computation and storage technologies. With this architecture, physical node resources can be upgraded with transport technologies independent of the evolution of computing and storage resources, and vice versa. It also facilitates scaling of transport resources regardless of computing and storage resources, and vice versa.

9 Physical resource management functions

Physical resource management functions manage, control, and monitor physical resources, in collaboration with virtual resource management functions. They include physical resource configuration functions, physical resource monitoring and fault management function, and, optionally, physical resource discovery function.

When physical resource management functions are independently implemented for different layers on a physical link resource, there should be a communication channel between the functions of individual layers for consistent and coordinated management.

9.1 Physical resource configuration functions

Physical resource configuration functions are comprised of the following functions:

- **Abstraction function**. The abstraction function creates virtual resources by abstracting physical resources. In doing so, the function may divide a single physical resource into multiple physical resources or merge multiple physical resources into a single physical resource. It may also create logical resources, which are logically grouped physical resources. For example, a physical link can be divided into multiple logical links, where each logical link is represented by parameters, such as a link identifier (ID) and end-point addresses;
- **Allocation function**. The allocation function allocates one or more physical resources to one or more virtual resources upon receiving a request from virtual resource configuration functions;
- **De-allocation function**. The de-allocation function releases specific physical resource(s) being used for one or more virtual resources. De-allocation is initiated upon receiving a request from virtual resource configuration functions;
- **Re-allocation function**. The re-allocation function replaces specific physical resource(s) being used for one or more virtual resources with other physical resources, thus allocating the latter physical resources to the same virtual resources. Re-allocation may be initiated upon receiving a request from virtual resource configuration functions or independently from them.

9.2 Physical resource monitoring and fault management function

Physical resource monitoring and fault management function monitors and collects status, performance, and other kinds of statistics of physical resources. The function also detects performance degradations, failures, and other kinds of anomalies of physical resources. When that happens, the function identifies the causes and takes procedures to deal with the problem. For example, the function may initiate re-allocation of specific physical resource(s) using physical resource configuration functions.

9.3 Physical resource discovery function

Physical resource discovery function detects new physical resources and reports them to virtual resource discovery function of virtual resource management functions.

10 Virtual resource management functions

Virtual resource management functions manage, control, and monitor virtual resources in collaboration with physical resource management functions, LINP management functions, and LINP operator functions. They include virtual resource configuration functions, virtual resource monitoring and fault management function, and, optionally, virtual resource discovery function.

When virtual resource management functions are independently implemented for different layers on a virtual link resource, there should be a communication channel between the functions of individual layers for consistent and coordinated management.

10.1 Virtual resource configuration functions

Virtual resource configuration functions are comprised of the following functions:

- **Allocation function**. The allocation function allocates specific virtual link resources and virtual node resources to an LINP upon receiving request from resource coordination function of LINP management functions;

- **De-allocation function**. The de-allocation function releases specific virtual link resources and/or virtual node resources being used for a LINP. De-allocation is initiated upon receiving request from resource coordination function of LINP management functions.

10.2 Virtual resource monitoring and fault management function

Virtual resource monitoring and fault management function monitors and collects status, performance, and other kinds of statistics of virtual resources. The function also detects performance degradations, failures, and other kinds of anomalies of virtual resources. When that happens, the functions identify the causes and take procedures to deal with the problem. For example, the functions may initiate de-allocation of specific virtual resources using virtual resource configuration functions.

10.3 Virtual resource discovery function

Virtual resource discovery function detects new virtual resources and reports them to resource coordination function of LINP management functions.

11 LINP management functions

LINP management functions manage, control, and monitor LINPs, in collaboration with virtual resource management functions, LINP operator functions, and service deployment functions. They include resource orchestration function, LINP configuration functions, LINP monitoring and fault detection function, and authorization functions.

11.1 Resource coordination function

Resource coordination function chooses specific virtual resources among available ones to constitute a LINP, and create a mapping between them. For this purpose, the function examines properties of virtual resources, such as bandwidth and memory size that reflects properties of constituent physical resources, and coordinate mappings.

Resource coordination function may change an already created mapping, when needed or appropriate. For example, when a malfunctioning virtual resource is detected, or when a new virtual resource is detected, the mapping involving it can be changed.

11.2 LINP configuration functions

LINP configuration functions enable service deployment functions to deploy service on an LINP. Besides this, the functions configure a LINP in the following ways:

- **Binding**: Binding function binds virtual link and virtual node resources to a LINP to which the virtual resources are allocated, upon receiving a request from LINP operation functions or from service deployment functions. As a result of binding, the LINP becomes ready to be activated;
- **Unbinding**: Unbinding function unbinds virtual link and virtual node resources from an LINP to which the virtual resources are bound, upon receiving a request from LINP operation functions or from service deployment functions;
- **Service activation**: Service activation function starts providing service on a LINP. Examples of service activation include running virtual machines (VMs) and enabling gateway functions. In this process, the functions may examine whether the LINP is functioning properly. Service activation is initiated upon receiving request from LINP operator functions or from service deployment functions. Service activation has to be preceded by binding;
- **Service deactivation**: Service deactivation function stops providing service on a LINP. Examples of service deactivation include stopping VMs and disabling gateway functions. Service activation is initiated upon receiving request from LINP operator functions or from service deployment functions. Service deactivation has to be preceded by unbinding.

11.3 LINP monitoring and fault detection function

LINP monitoring and fault detection function monitors and collects status, performance, and other kinds of statistics of LINPs. The function also detects performance degradations, failures, and other kinds of anomalies of LINPs. When that happens, the function notifies the detected events to virtual resource management functions to deal with the problem.

11.4 Authorization functions

Authorization functions provide the following:

- **LINP operator authorization**: LINP operation authorization function authorizes LINP operators to access LINPs. The function can also be used to authorize LINP exchangers;
- **Service developer authorization**: Service developer authorization function authorizes service developers to access LINPs;
- **User terminal access administration**: User terminal access administration function administers information for controlling accesses from end-users to LINPs.

12 LINP operator functions

LINP operator functions operate services on LINPs, in collaboration with virtual resource management functions and LINP management functions.

To enable or disable service on a LINP, the functions send a request of service activation or of service deactivation to LINP management functions, respectively.

While operating services, LINP operator functions can also monitor the status of virtual resources by accessing virtual resource monitoring and fault management function.

13 Service deployment functions

Service deployment functions, implemented either by a network operator or by a service developer, manage, control, and operate services on LINPs, in collaboration with LINP management functions and service developer functions. When the functions are implemented by a service developer, the functions use the LMI reference point to access LINP management functions as shown in Figure 9.

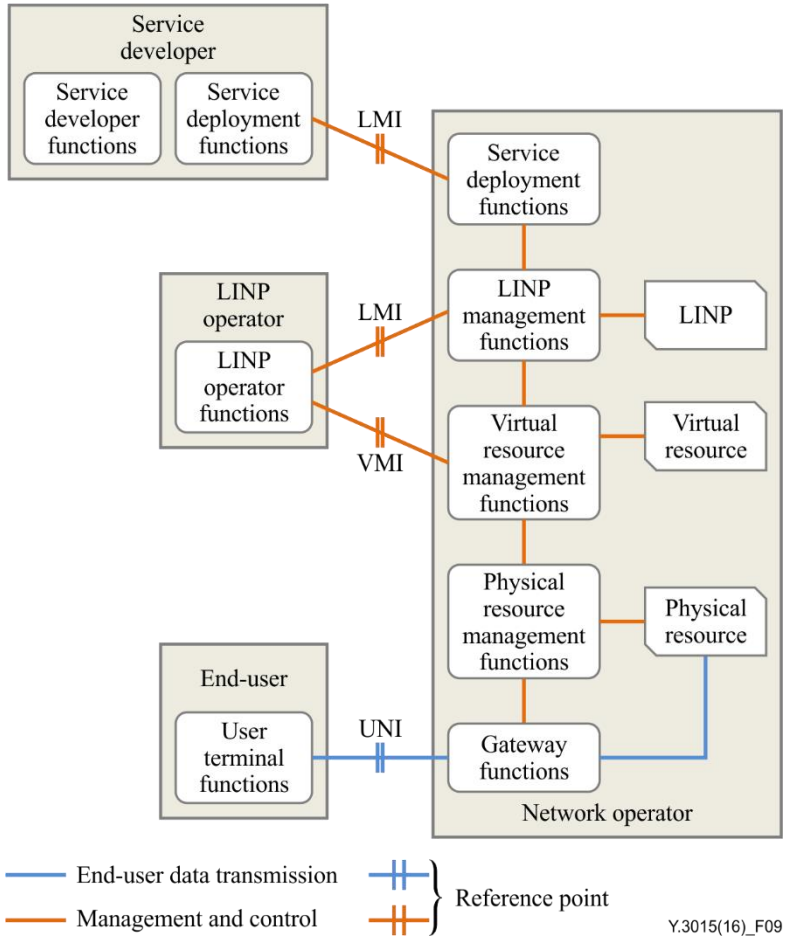

Figure 9 – Functional architecture with service deployment functions supported by a service developer

Service deployment functions deploy services, which are programmed by service developer functions, on a LINP. They collect parameters of the LINP by communicating with LINP management functions in order to deploy the service on the LINP automatically. To enable or disable a service on the LINP, the functions send a request for service activation or for service deactivation, respectively, to LINP management functions.

Service deployment functions can be used to provide an on-demand service to an end-user, in which case they discover an appropriate LINP that is highly suitable for the service. In addition, when the attachment point is changed by the end-user, or when the LINP on which the on-demand service is running does not have enough resources, the on-demand service is reallocated to another appropriate LINP by service deployment functions. When the service is reallocated, the set of virtual resources being used to support the service and the end-user sessions associated with it should be moved together to maintain service continuity by avoiding any downtime.

Service deployment functions can also be used for coordination of services. For example, they may manage deployment of applications that together constitute a service across multiple administrative domains of the same network operator.

14 Service developer functions

Service developer functions enable service developers to design specific services to be deployed on a LINP. The functions combine service components, which have been developed in advance and can be reused. Service elements may consist of widely used data processing functions, like data

forwarding, data routing, and media transcoding. They can also include additional data processing functions, such as data or packet caching and packet flow processing.

15 Gateway functions

Gateway functions are used to connect end-users to LINPs. Gateway functions may also provide authentication and authorization functions for end-users to access LINPs. Figure 10 illustrates the gateway functions and their interactions with the LINP management functions.

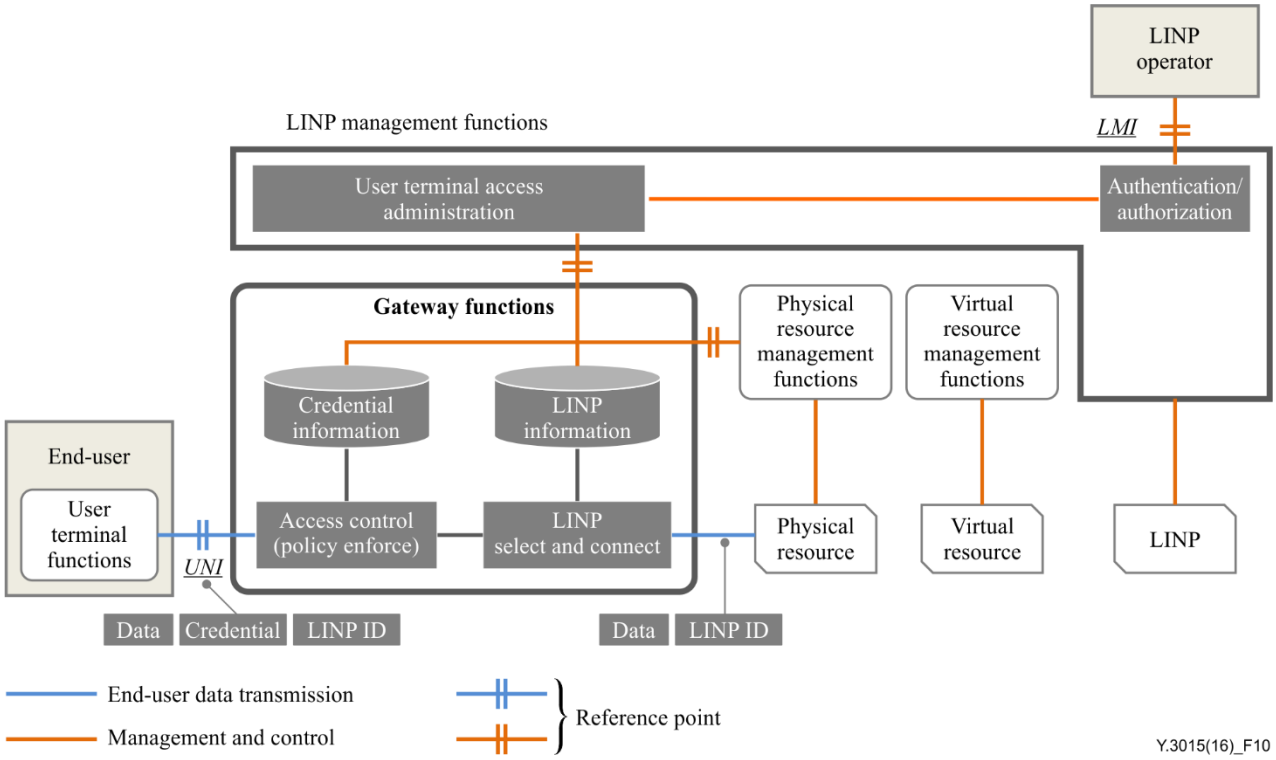

Figure 10 – Gateway functions in relation to LINP management functions

Through the UNI reference points, end-users' data are exchanged between LINPs and end-users. The end-users' data frames/packets through the UNI reference points should include the following information:

- End-users' data;
- LINP IDs or correspondent IDs to uniquely identify the source LINP and the destination LINP;
- Credentials for access control (optional).

Through LMIs, access policies of LINPs for individual end-users are configured. For this purpose, the following information should be provided at the LMI:

- Service developers' credentials;
- LINP access policies for end-users.

Through LMIs, end-users' credentials for accessing LINPs are delivered. For this purpose, the following information should be provided at the LMI:

- End-users' credential.

Gateway functions should include the following functions.

– Access control (policy enforcement) function, which determines if an end-user's data frames/packets are allowed to go through a specified LINP by checking the credential in the end-user's data frames/packets;

– LINP select function, which extracts LINP IDs or correspondent IDs from end-users' data frames/packets and identify destination LINPs;

– LINP connect function, which makes connections between end-users and LINPs with protocol translation if it is needed.

Gateway functions may provide internal reference point for gateway control within an administrative domain. As shown in Figure 10, the internal reference point resides between gateway functions and LINP management functions. It is used to deliver the following information from LINP management functions to gateway functions:

– Information for controlling accesses from end-users to LINPs;

Relationship between end-users and the LINPs.

– Another internal reference point may reside between gateway functions and physical resource functions. The data frames/packets through this internal reference point should include the following data:

– End-users' data;

– LINP ID or correspondent ID from which the destination LINP is uniquely specified.

16 User terminal functions

User terminal functions connect end-users to LINPs through gateway functions. User terminal functions may also implement authentication functions. Figure 11 illustrates the user terminal functions and the interaction with other functions.

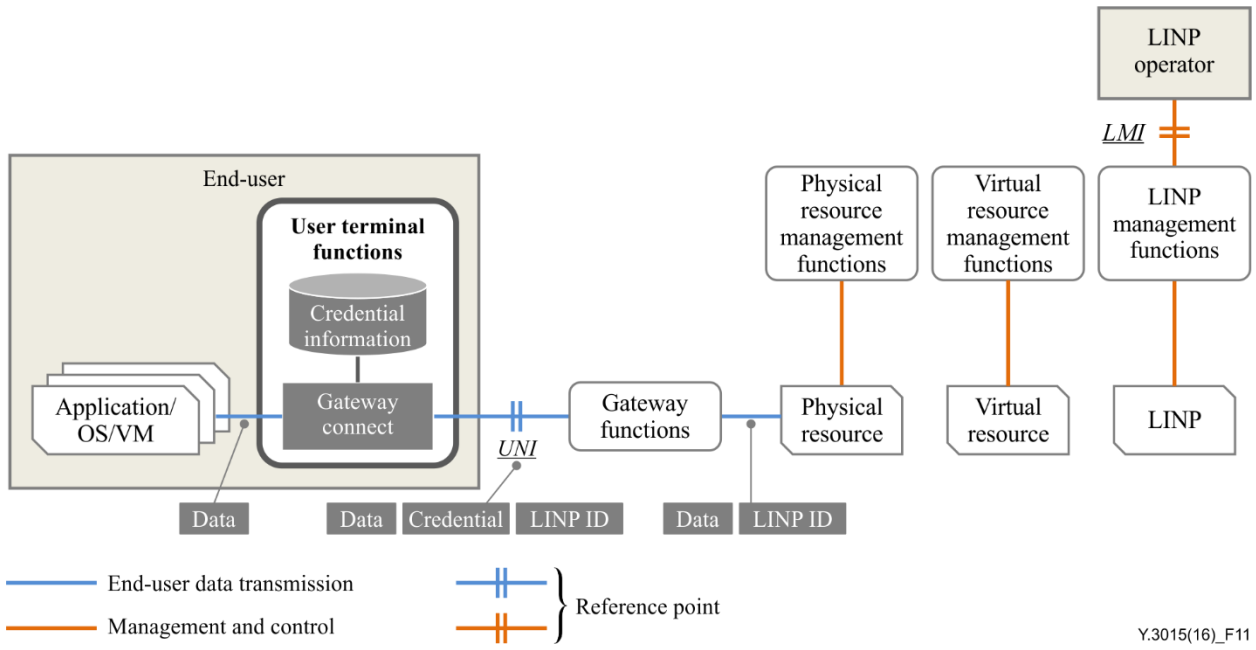

Figure 11 – User terminal functions in relation to gateway functions

Through the UNI reference points, end-users' data are exchanged between gateway functions and user terminal functions. The data frames/packets should include the following information:

– End-users' data;

- LINP IDs or correspondent IDs to uniquely identify the source LINP and the destination LINP;
- Credentials for authentication to connect to gateway functions (optional).

In order to provide the functions described above, user terminal functions should include the following function:

- Gateway connect function, which adds LINP IDs or correspondent IDs to end-users' data frames/packets for specific applications, operating systems (OSs), or VMs in a user terminal. The function also extracts LINP IDs or correspondent IDs from end-users' data frames/packets and identifies destination applications, OSs, or VMs in the user terminal.

If the gateway connect function adds an LINP ID to all data frames/packets from applications, OSs, or VMs in a user terminal, the user terminal is attributed to a single LINP. On the other hand, if gateway connect function adds a different LINP ID per application, OS, or VM, each application, OS, or VM is attributed to a different LINP.

The information needed to associate LINP IDs or correspondent IDs with specific applications, OSs, or VMs in a user terminal may be provided by a service developer to end-users.

17 Federation functions

Federation functions are required if LINPs need to be expanded across multiple administrative domains. Federation functions allow multiple VMIs for management and control and multiple NNIs for forwarding end-users' data coming from different administrative domain.

To achieve this, federation functions have to reconcile the differences in specifications of NNIs and/or VMIs reference points, LINPs, and so on. Followings are the functions used for this purpose:

- Command translation: Each administrative domain may have different commands for LINP creation and management. In that case, command translation function is required to translate each administrative domain's commands and parameters into another administrative domain's commands and parameters;
- State transition mapping: Each administrative domain may have different state transition for LINPs, virtual node resources and virtual link resources. In that case, state translation function is required to maintain the difference in state during application programming interface (API) conversion;
- Policy translation: Each administrative domain may have different policies for LINP creation and management. In that case, policy translation function is required to reconcile the differences;
- LINP definition translation: Each administrative domain may have different formats to define LINP configurations. In that case, LINP definition translation function is required to convert an LINP definition of one administrative domain into that of another administrative domain;
- Data plane conversion: Each administrative domain may have different end-users' data packet formats. In that case, data plane conversion function is required to convert a packet format of one administrative domain into that of another administrative domain;
- Protocol/resource negotiation: Each administrative domain may have different capabilities in handling protocols and resources. Federation functions are recommended to include protocol/resource negotiation function. This function may be used when failures occur or new resources are detected. In that case, the function is executed before federating LINPs.

18 Reference points

The reference points defined in this Recommendation are given in the following clauses.

18.1 User-to-network interface (UNI)

A UNI reference point resides between gateway functions of a network operator and user terminal functions of an end-user.

18.2 Network-to-network interface (NNI)

An NNI reference point resides between physical resources that individually belong to two different network operators. It can also reside between physical resources of a network operator and federation functions of an LINP exchanger when LINP federation is done using the LINP exchanger's federation functions.

18.3 Virtual resource management interface (VMI)

A VMI reference point resides between virtual resource management functions of a network operator and LINP operator functions of an LINP operator.

18.4 LINP management interface (LMI)

An LMI reference point resides between LINP management functions of a network operator and LINP operator functions of an LINP operator. It can also reside between LINP management functions of a network operator and service deployment functions of a service developer when service deployment functions are implemented by the service developer.

18.5 Service management interface (SMI)

An SMI reference point resides between service deployment functions of a network operator and service developer functions of a service developer when service deployment functions are implemented by the network operator.

18.6 Programmer-to-redirector interface (PRI)

A PRI reference point resides between a programmer and a redirector of physical node resource.

19 Security considerations

To make sure that all the LINPs are logically independent, virtual resources allocated to each LINP have to be isolated from those allocated to other LINPs. From the viewpoint of security, isolation of virtual resources is essential in preventing parties from having access or influence to unauthorized LINPs, whether or not they have a malicious intention. Thus, abstraction and allocation that physical resource management functions execute have to ensure the isolation of individual virtual resources. In addition, it is desirable that abnormal use of virtual resources can be detected by a collaborative mechanism of physical resource monitoring and fault management functions, virtual resource monitoring and fault management functions, and LINP monitoring and fault detection functions.

Also important for security is the implementation of authentication and authorization mechanisms. Since various kinds of user roles are involved in an LINP, network operators have to execute authentication and authorization for each kind of user roles. End-users that wish to enjoy services have to be authorized before they become able to access the corresponding LINPs. Service developers and LINP operators also have to be authorized before becoming able to execute their functions through appropriate reference points. LINP management functions and gateway functions that network operators implement are responsible in this regard.

Appendix I

Implementation example of network virtualization

(This appendix does not form an integral part of this Recommendation.)

This appendix provides an implementation example of physical node architecture. The example is based on an implementation model of 'VNode system,' which is a network system developed to realize programmable virtual networks [b-VNode Whitepaper]. Adopting the node architecture presented in clause 9, VNode system enables to incorporate software-defined networking (SDN) and Network Functions Virtualization (NFV) technologies in an integrated manner.

Figure I.1 shows main part of the implementation model of the node architecture. The model includes VNode, a domain controller, physical and virtual links, and several interfaces corresponding to different reference points. A VNode is a component where physical resources and virtual resources reside. A domain controller is a component that controls multiple VNodes with management functions for physical resources, virtual resources, and LINPs.

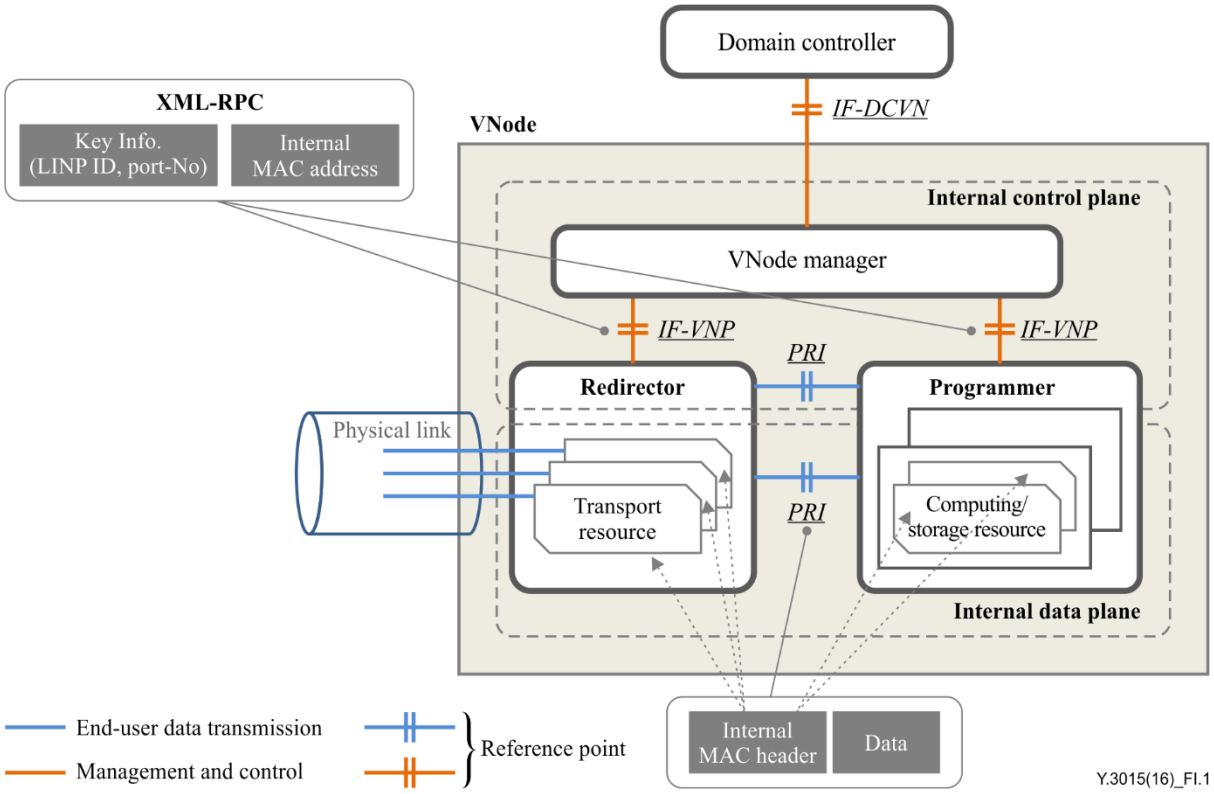

Figure I.1 – VNode system components and interfaces

A VNode represents a physical node and consists of a redirector, a programmer, and a VNode manager. As described in clause 9, a redirector contains transport resources while a programmer contains computing and storage resources.

A programmer utilizes various types of processing devices and/or software components. For instance, network processors and virtual machines (VMs) on general purpose servers can be used. They may be in a single physical structure or in physically separated structures. LINP operators can select and combine these kinds of components so as to meet requirements for programmability and performance. It should be noted that virtualized network functions (VNFs), which are key ingredient in NFV technologies, provide an instantiation example of programmability realized by a programmer.

A redirector provides both physical and virtual links that define structures of a virtual network. For instance, traditional switches and routers or OpenFlow switches can be used. It plays a central part in separating individual virtual links logically. The logical separations in the redirector are done based on mappings between virtual links to physical links at both outside and inside the VNode. It also conducts conversion of end-user data representations, such as data packet formats, between outside and inside the VNode. It should be noted that SDN technologies can fit well for the functionalities of redirectors.

A VNode manager controls both the programmer and the redirector in a same VNode. It implements functions delegated by a domain controller, the primal component in managing physical resources, virtual resources, and LINPs that are provided by a network operator.

For the sake of clarity, components in a VNode are classified logically as belonging to two planes, namely, internal control plane and internal data plane. A VNode manager executes functions of the internal control plane, while a programmer and a redirector execute functions for both the planes.

Several interfaces are defined for VNode system. As described in clause 18, PRI resides between the programmer and the redirector in a VNode. In accordance to separation of internal logical planes, specifics of PRI are defined for both the internal control and the internal data planes. For instance, Extensible Markup Language-Remote Procedure Call (XML-RPC) is defined for the internal control plane, while physical specifications are defined for the internal data plane. For both the planes, mac addresses of internal components are designated.

VNode system has additional interfaces. Interface for VNode manager-to-Programmer (IF-VNP) and Interface for VNode manager-to-Redirector (IF-VNR) are defined for the internal control plane of a VNode. IF-VNP resides between a VNode manager and a programmer, while IF-VNR between a VNode manager and a redirector. Outside VNodes, Interface for Domain Controller-to-VNode (IF-DCVN) is defined between a domain controller and a VNode manager of each of the VNodes. It is used for management and control, and, as such, is understood as containing both VMI and LMI.

Figure I.2 depicts an example of virtual network and its implementation using VNode system. In this example, two LINPs are provided on top of a virtual network infrastructure. Each LINP is provided using a different programmer at some VNodes. This kind of implementation can happen when very different QoS are required for individual LINPs.

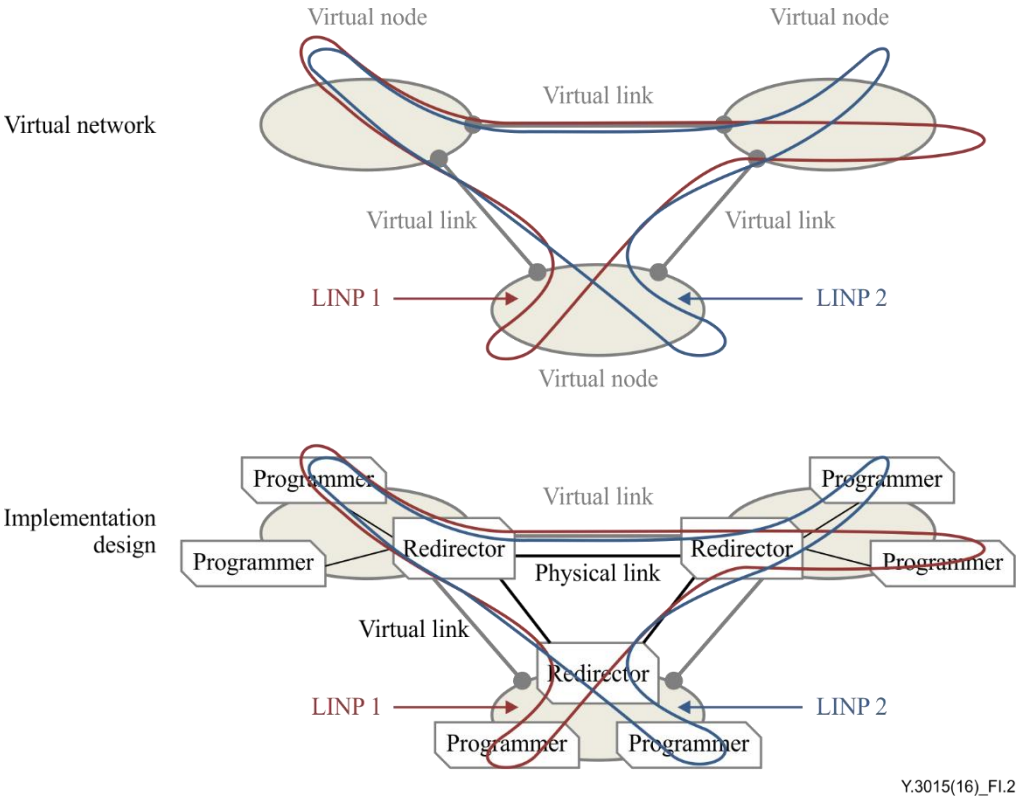

Figure I.2 – An example of virtual network and its implementation using VNode system

Bibliography

[b-ITU-T Y.3502] Recommendation ITU-T Y.3502 (2014), *Information technology – Cloud computing – Reference architecture*.

[b-ITU-T Y.3511] Recommendation ITU-T Y.3511 (2014), *Framework of inter-cloud computing*.

[b-VNode Federation] VNode Project (2014), *Federation Architecture and Common API / Common Slice Definition*, http://nvlab.nakao-lab.org/Common_API_V2.0.pdf

[b-GENI Architecture] GENI Design Document (2007), *Overview of the GENI Architecture*, http://groups.geni.net/geni/raw-attachment/wiki/OldGPGDesignDocuments/GDD-06-11.pdf

[b-VNode Whitepaper] VNode Project (2014), *White Paper*, https://nvlab.nakao-lab.org/vnode-white.paper.pdf

SERIES OF ITU-T RECOMMENDATIONS

Series A	Organization of the work of ITU-T
Series D	General tariff principles
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	Cable networks and transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Environment and ICTs, climate change, e-waste, energy efficiency; construction, installation and protection of cables and other elements of outside plant
Series M	Telecommunication management, including TMN and network maintenance
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Terminals and subjective and objective assessment methods
Series Q	Switching and signalling
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	Data networks, open system communications and security
Series Y	**Global information infrastructure, Internet protocol aspects and next-generation networks, Internet of Things and smart cities**
Series Z	Languages and general software aspects for telecommunication systems

Printed in Switzerland
Geneva, 2016