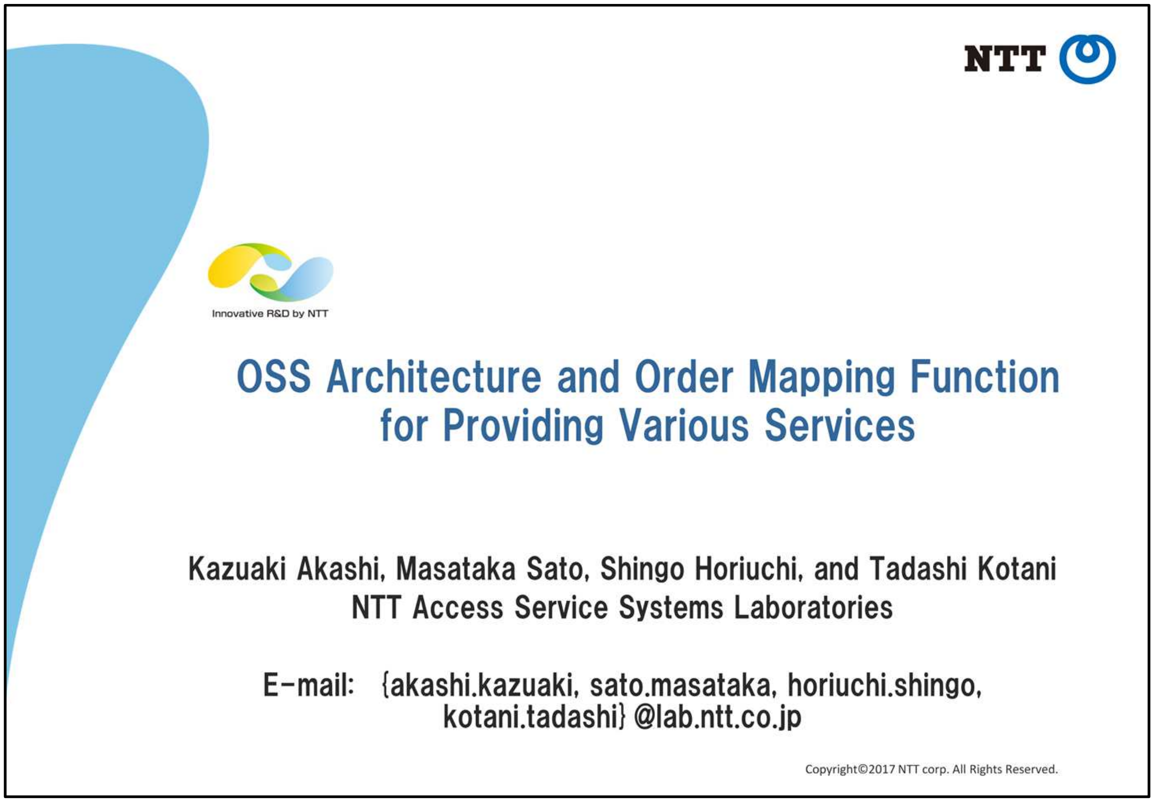
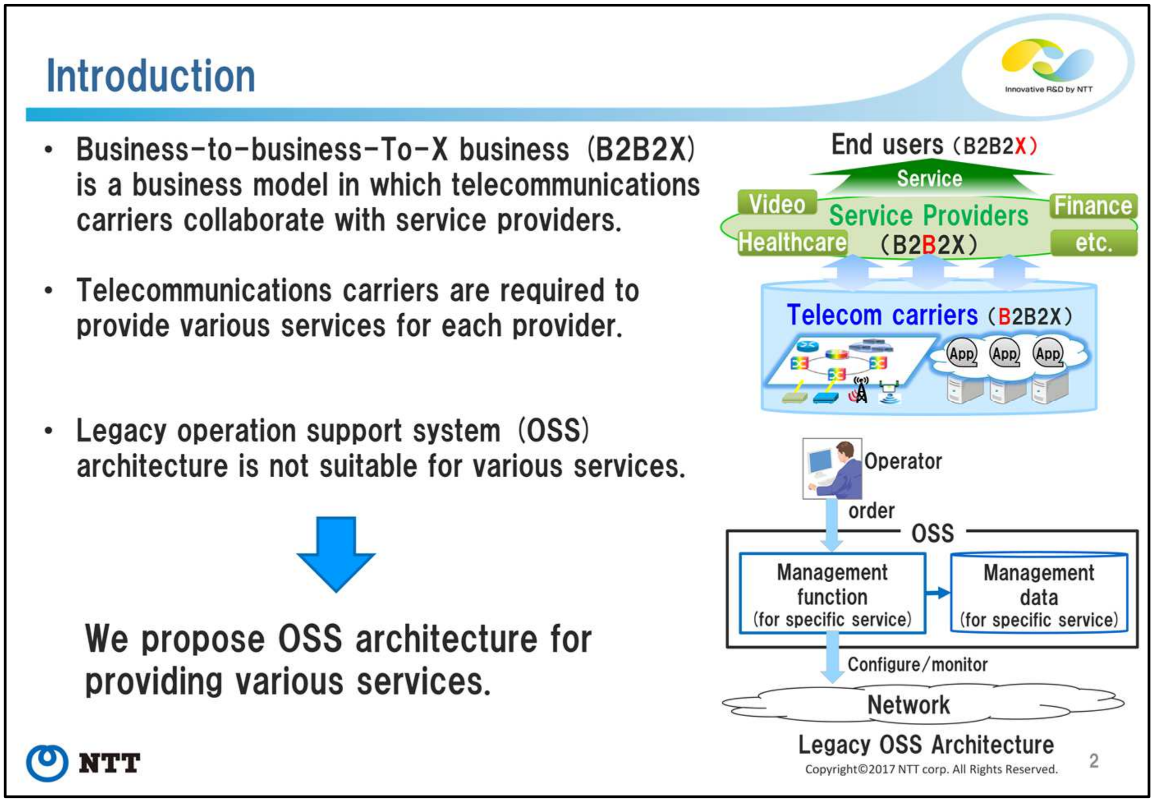


Abstract

Telecommunications carriers are required to provide various services according to service-provider requirements in the business-to-business-to-X business model, or B2BX. However, an architecture of a legacy operation support system (OSS) is not suitable for providing various services because of high development cost to add such services. In this paper, we propose an OSS architecture that can easily provide various services. Our proposed architecture can suppress the development cost by the external definition of the processing difference for each type of service and loose coupling of the service and resources. We also investigated management data and functions to develop an OSS based on the proposed architecture. We use an information framework standardized by TM Forum (TMF) and propose an order-mapping function that can flexibly change the combination of services and resources.

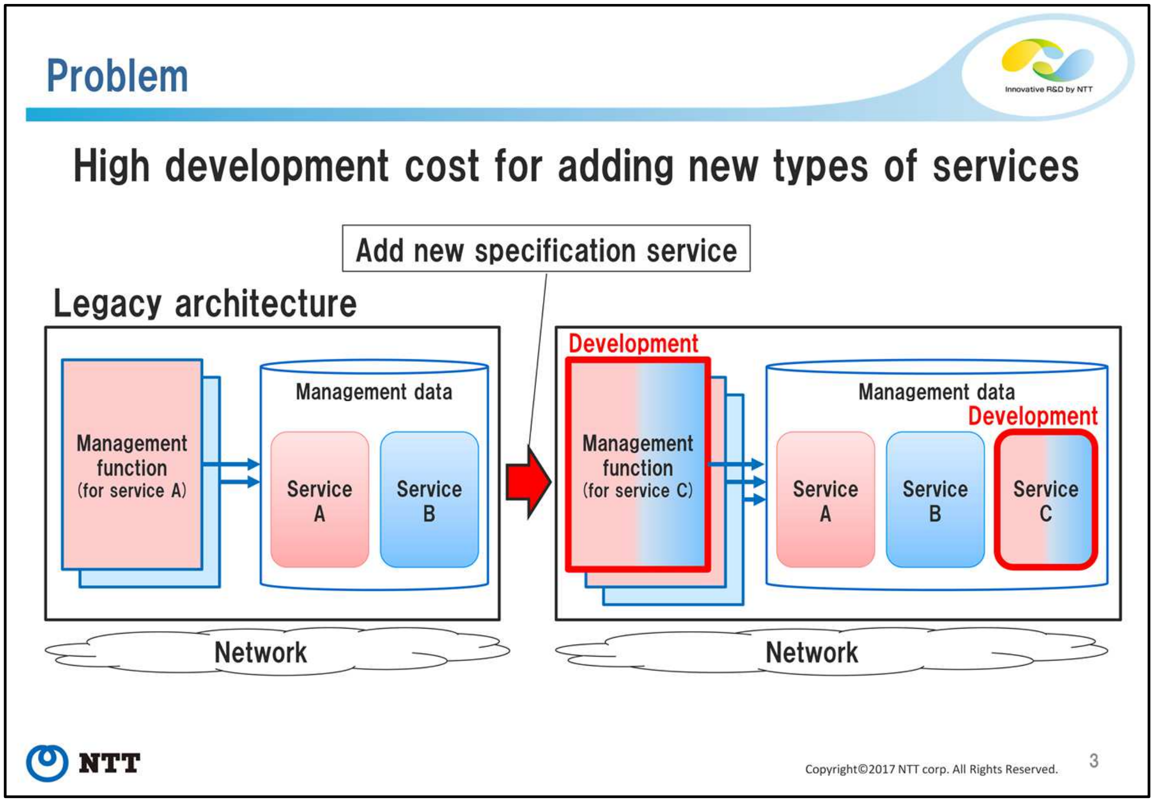


1. Introduction

Business-to-business-to-X (B2B2X) is a business model in which telecommunications carriers collaborate with service providers, where B means telecommunications carriers, middle B means service providers, and X means end users such as customers, companies, and governments. Telecommunications carriers provide network and cloud services to service providers, and service providers provide the value-added services to end users.

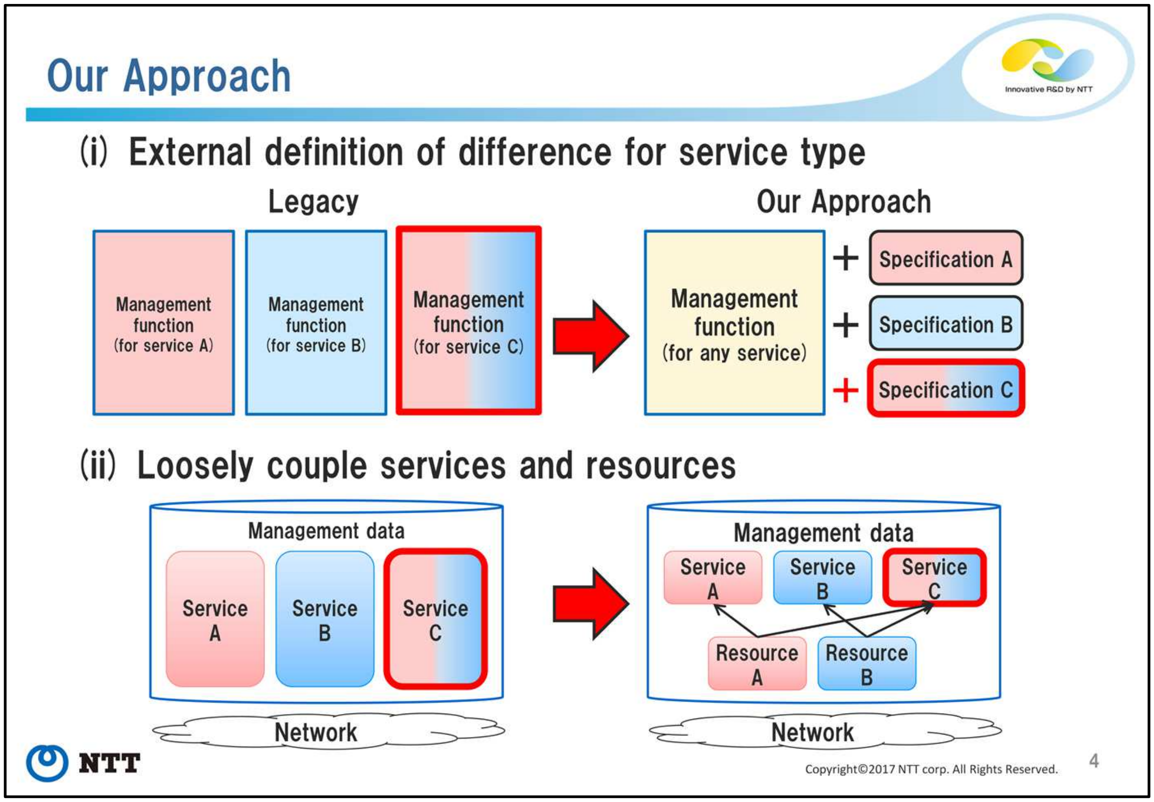
In B2B2X business, telecommunications carriers are required to provide various services according to the requirements of each service provider. The service provider’s requirements differ depending on the industry and company. However, the architecture of a legacy operation support system (OSS) is not suitable for providing various services. In such a legacy architecture, management functions and management data are individually optimized for each service. Therefore, when adding a new type of service, the telecommunications carrier has to develop new management functions and data.

In this paper, we propose an OSS architecture for providing various services. The proposed architecture supports development of an OSS that facilitates the addition of new types of services. To develop an OSS based on the proposed architecture, we use an information framework standardized by the TM Forum [1] and propose an order-mapping function that can flexibly change the combination of services and resources.



2. Problem

A legacy OSS architecture incurs high development cost as the types of services increase. In many cases, telecommunications carriers develop OSSs according to the services, so their functions and data are individually optimized for specific services. Therefore, the legacy OSS architecture has management functions and management data for each type of service. When adding a new type of service in the legacy OSS architecture, it is necessary to develop a new management function and management data, as shown in this figure, which requires high development cost. Also, if existing management functions and management data are re-modeled, their functions and data may be complicated.

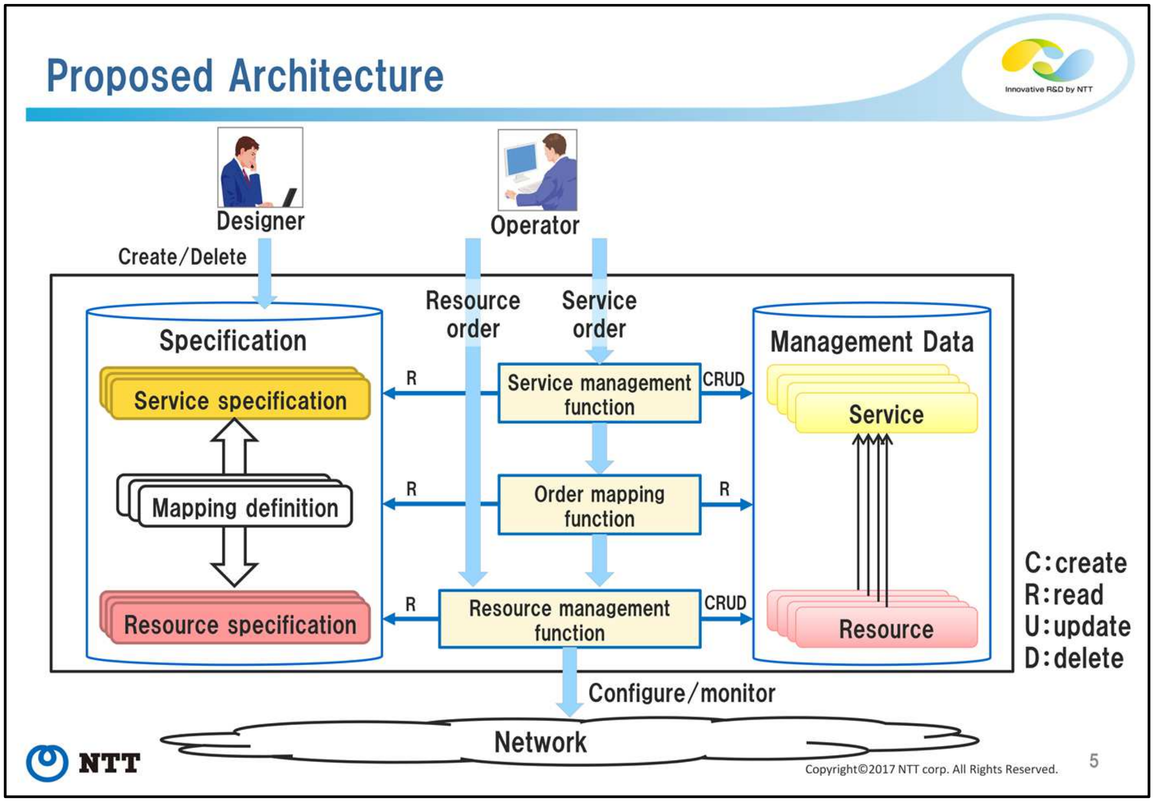


3. Approach

There are two approaches to lower the OSS development cost associated with an increase in the types of services.

The first is to externally define the processing differences for each service type. In the legacy architecture, there are management functions for each type of service. However, similar processing is implemented in these management functions. For example, when parameters to be set in a communication device in a network are determined using parameters included in management data, though the parameters to be used are different for each service type, the processing in each management function is probably similar. The management function can be shared by externally defining the processing difference depending on the service type as a specification. This allows telecommunications carriers to add new service types simply by adding specifications without developing management functions.

The second is loosely coupling services and resources. Management data individually optimized for each service likely include not only service information but also resource information such as about communication devices and communication paths in a network for providing the service. By separately management data on services and resources, it is thought that various services can be provided by changing the combination of resources. For services offered using existing resource types, new service types can be added without adding resource types.



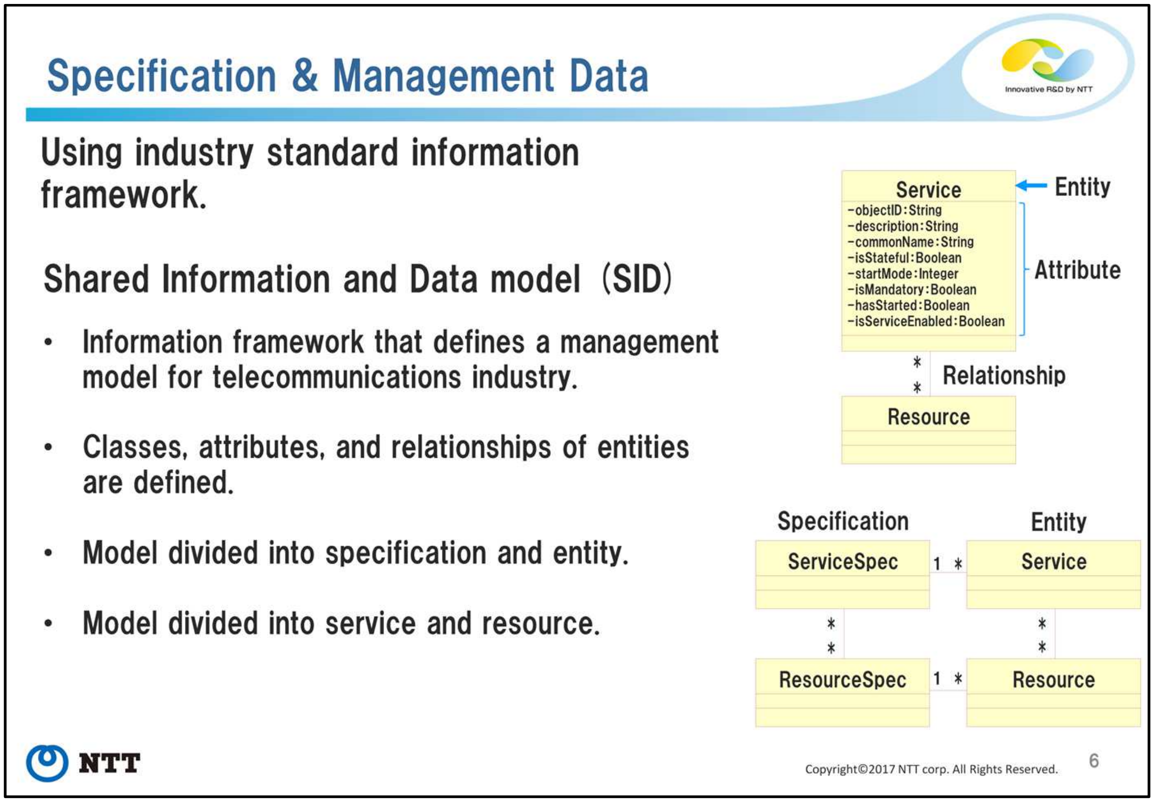
4. Proposed Architecture

Based on the above two approaches, our OSS architecture can easily provide various services.

The proposed architecture has specifications created for each service and resource type, shared management functions independent of these types, and management data. There are three types of specifications, service, resource, and mapping, which are respectively created or deleted by service designers. The mapping definition defines relationships between service and resource specifications and defines the types and quantities of resources required when a certain type of service is requested. Management functions process orders according to these specifications.

In the proposed architecture, service- and resource-management functions and data are loosely coupled by the order-mapping function. The order-mapping function translates a service order, which requests any kind of service, into resource orders, which requests resources for the service. First, the service-management function receives a service order from the operator, creates/deletes/updates management data based on the service specification, and distributes the service order to the order-mapping function. The order-mapping function translates the service order received from the service-management function into the resource order based on the mapping definition and management data. Finally, the resource-management function creates/deletes/updates management data based on the resource specification and sets the device in the network by using these resource-management data.

To add a new service that combines existing resources, the service designer can add a service specification and mapping definition. Therefore, development of new management functions and resource specifications are not necessary. If a new resource type is required, it is necessary to add a resource specification, but development of the management function is unnecessary.



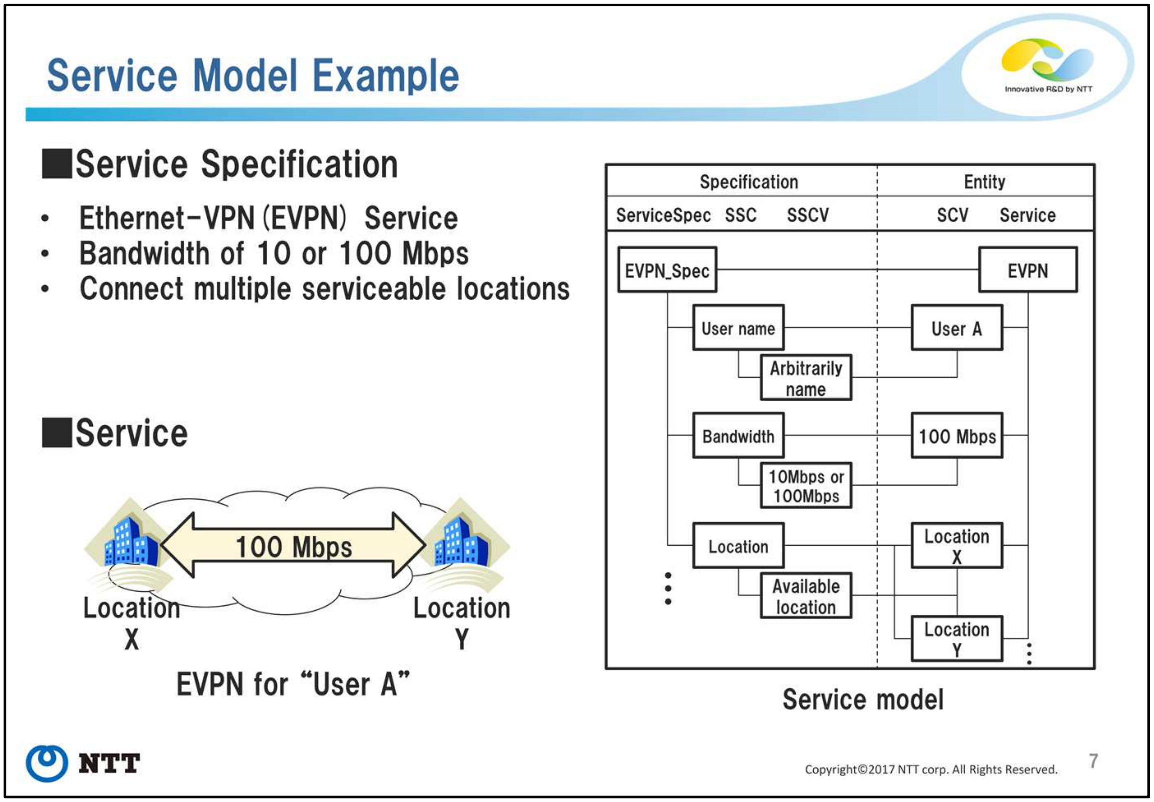
5. OSS based on the proposed architecture

We introduce specifications, management data and our proposed order-mapping function to develop an OSS based on our architecture.

5.1. Specification & Management data

We are considering specifications and management data by using an information framework standardized by the TM Forum. The information framework is the Shared Information and Data model, or SID [2], which defines management models common to the telecommunications industry. The SID is useful for OSS development based on our proposed architecture.

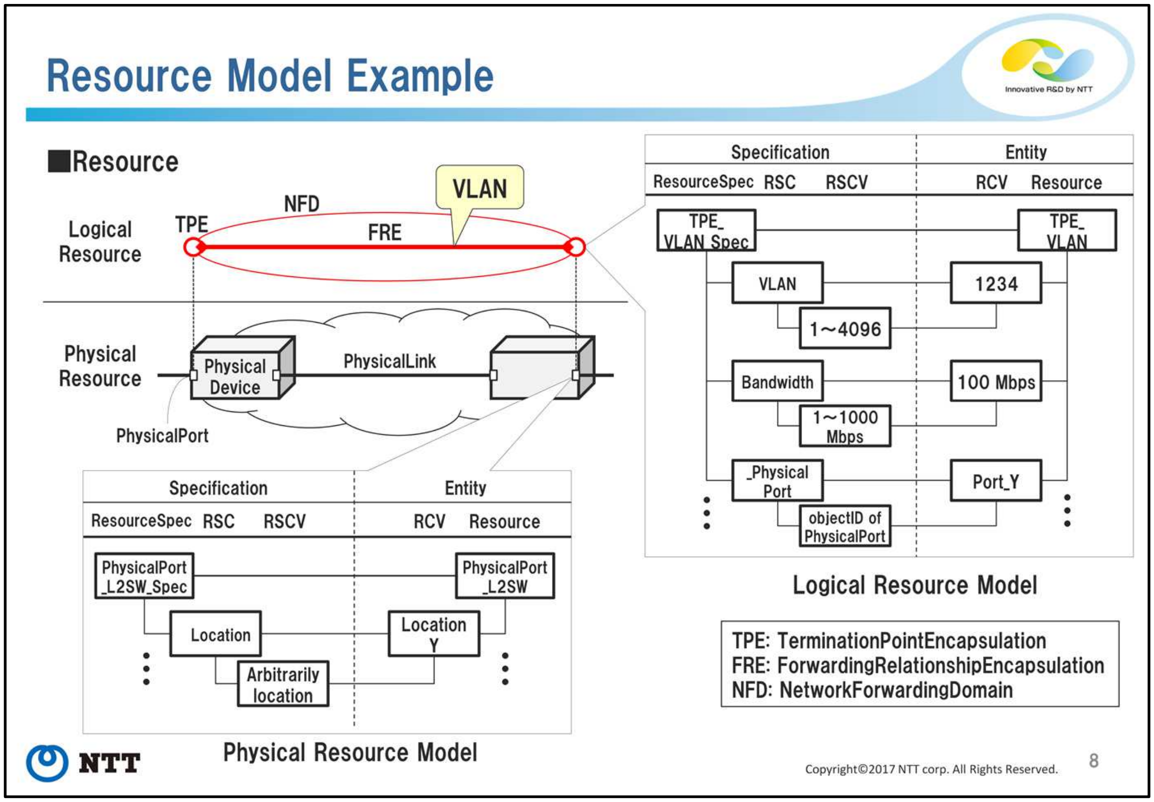
By modeling with the SID, services, resources, and their specifications can be expressed in a unified form. The management models in the SID include entity classes, their attributes, and relationships between classes. An entity means a managed object such as a service and resource. As shown in this figure, an entity and its specifications classes, service, and resource classes are defined separately in the SID. Modeling using the SID supports total optimization of management data since information of various managed objects and specifications can be organized using a common management model.



An example of a service model by using the SID is shown in this figure. This is an example of modeling the service and its specification to provide an Ethernet virtual private network, or EVPN, between locations specified by the user.

The specification of the service can be modeled using a "ServiceSpecification, or ServiceSpec", class, "ServiceSpecCaracteristic, or SSC", class representing the attribute, and "ServiceSpecCaracteristicValue, or SSCV", class representing the allowable values of attributes [3]. In this figure, "EVPN_Spec", which means a service type of EVPN modeled by "ServiceSpec", and attributes (user name, bandwidth, location), which characterize an EVPN modeled by the SSC, and its allowable values (arbitrary name, 10 or 100 Mbps, available location) modeled by the SSCV.

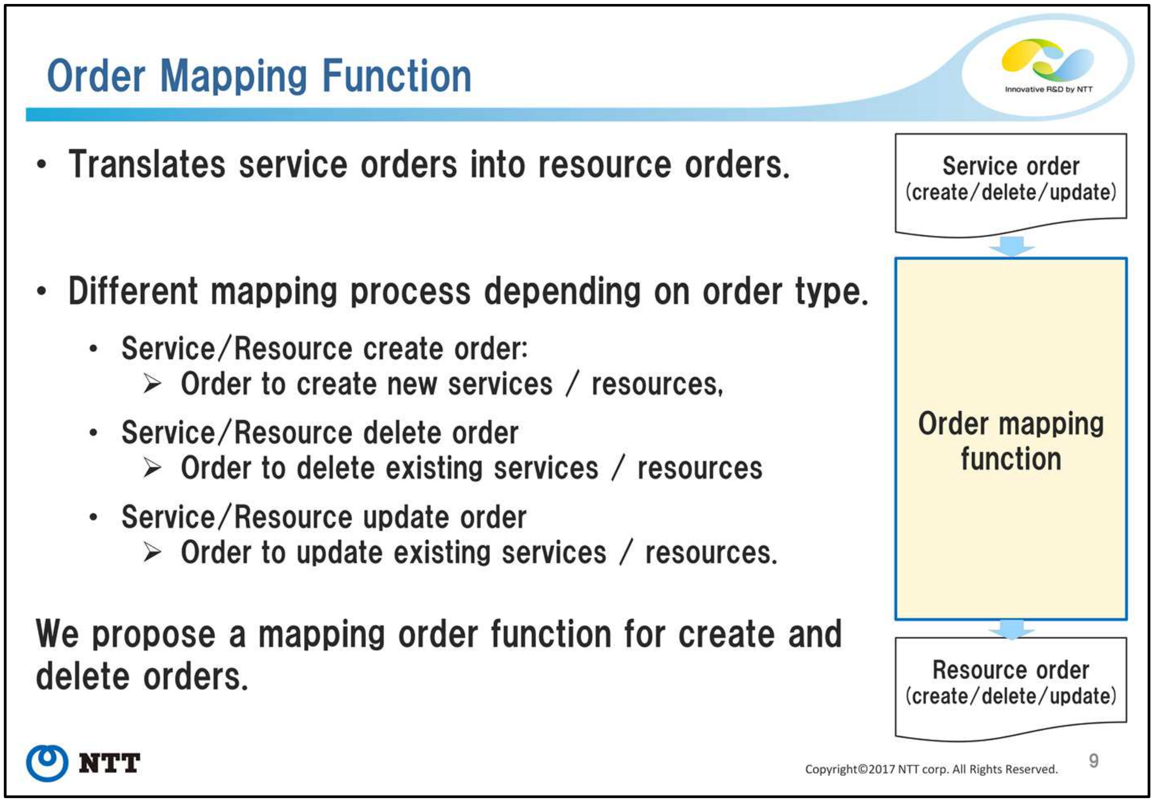
Similarly, the entity of the service can be modeled using the "Service" class and "ServiceCaracteristicValue, or SCV", class. An EVPN modeled by "Service" refers to "EVPN_Spec" and has values (user A, 100 Mbps, location x and y) for each SSC associated with "EVPN_Spec" in the SCV. For services other than those shown in the example, such as IP-VPN, it can be modeled in the same way.



Like the service model, an example of a resource model is shown in this figure. To provide an EVPN, consider an example of setting a virtual local area network, or VLAN, on a simple network with two layer 2 (L2) switches connected.

Resources include physical and logical resources. A physical resource is a resource of a physical layer such as a communication device, physical port, or cable. Physical resources are modeled by classes that inherit a PhysicalResource class such as a “PhysicalDevice”, “PhysicalPort”, or “PhysicalLink” [4], as shown in this figure. On the other hand, a logical resource is a resource of a logical layer, such as an end point of a network protocol, communication path, and domain, that can set paths. Logical resources are modeled by classes that inherit a “LogicalResource” class such as a “TerminationPointEncapsulation, or TPE”, “ForwardingRelationshipEncapsulation, or FRE”, and “NetworkForwardingDomain, or NFD” [5]. A VLAN is modeled as a FRE, endpoints of the VLAN are TPEs, and the area where the VLAN can be created is modeled as an NFD.

Specifications for each resource can be modeled using a “ResourceSpecification, or ResourceSpec,” class, “ResourceSpecCaracteristic, or RSC,” class, and a “ResourceSpecCaracteristicValue, or RSCV” class. For example, “PhysicalPort_L2SW” in the figure refers to “PhysicalPort_L2Switch_Spec” and has a value for an attribute such as “location” defined in the RSC. On the other hand, the “TPE_VLAN” created on the physical port refers to “TPE_VLAN_Spec” and has values for attributes such as “VLAN ID”, “Bandwidth”, and “_ PhysicalPort”.



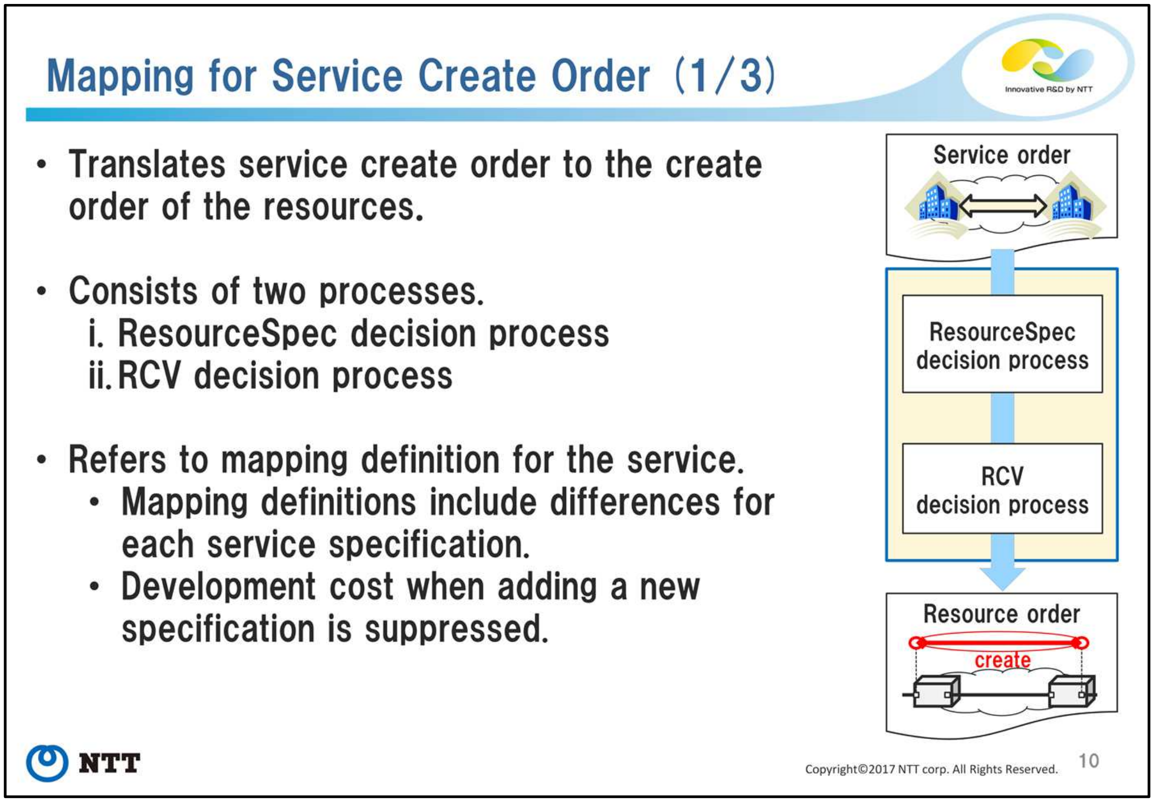
5.2. Order Mapping Function

5.2.1 Overview

The OSS based on the proposed architecture has an order-mapping function that translates service orders, which is inputted by operators to resource orders. The mapping process is dependent on the type of order. There are three types of orders for services and resources:

Service/Resource-create order: order to create a new service/resource,
Service/Resource-delete order: order to delete existing current service/resource,
Service/Resource-update order: order to update a current service/resource.

We propose an order-mapping function for create and delete orders, as explained in the following slides.



5.2.1 Mapping for Service Create Order

Based on the model described above, our proposed order-mapping function for a service-create order translates the order to create an order for resources needed for providing the service. The service-create order can be based on various specifications. The service-create order contains information about the service to be created, and the resource-create order contains information about the resource to be created.

There is a difference between the mapping process of different services. The proposed order-mapping function absorbs the difference by keeping the service-independent common part of the mapping process while enabling the defining of the service-dependent specification by mapping the definition function externally.

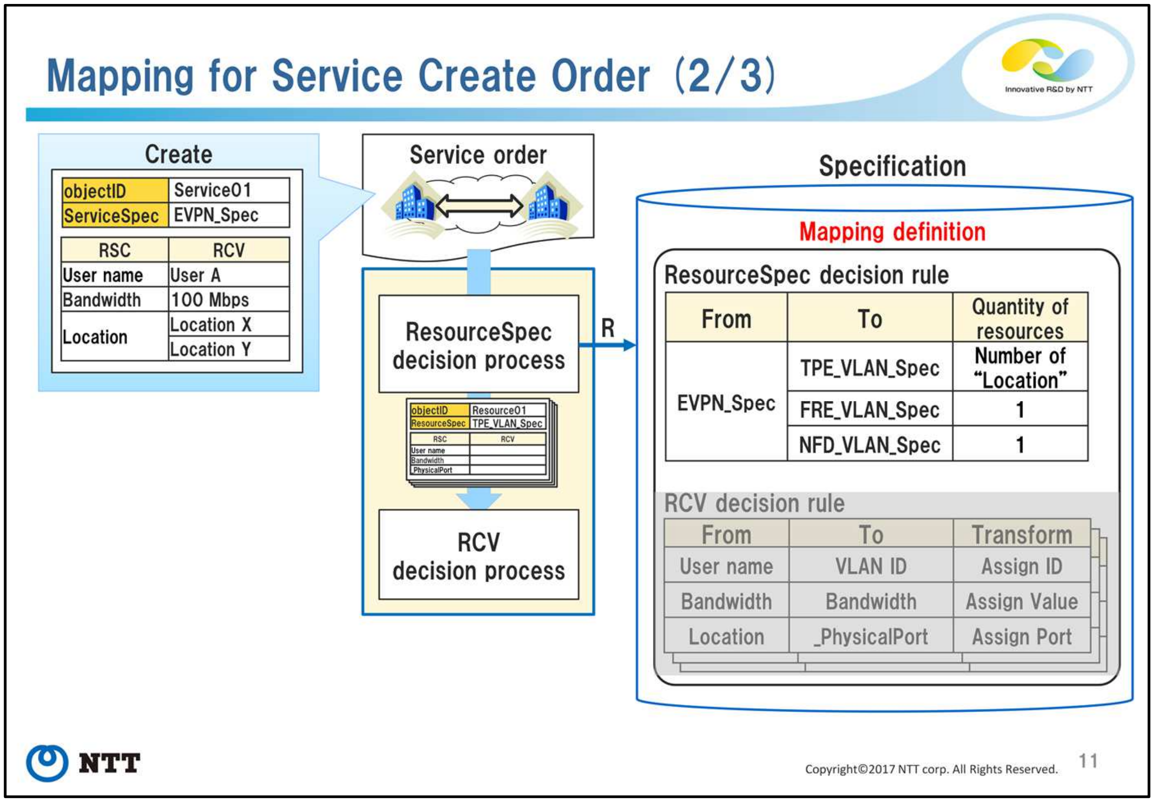
The proposed function consists of the following two processes that operate with reference to mapping definition.

i. ResourceSpec decision process

Process to determine specifications and quantity of resources for providing a service.

ii. RCV decision process

Process to determine the RCV of each resource determined in i.

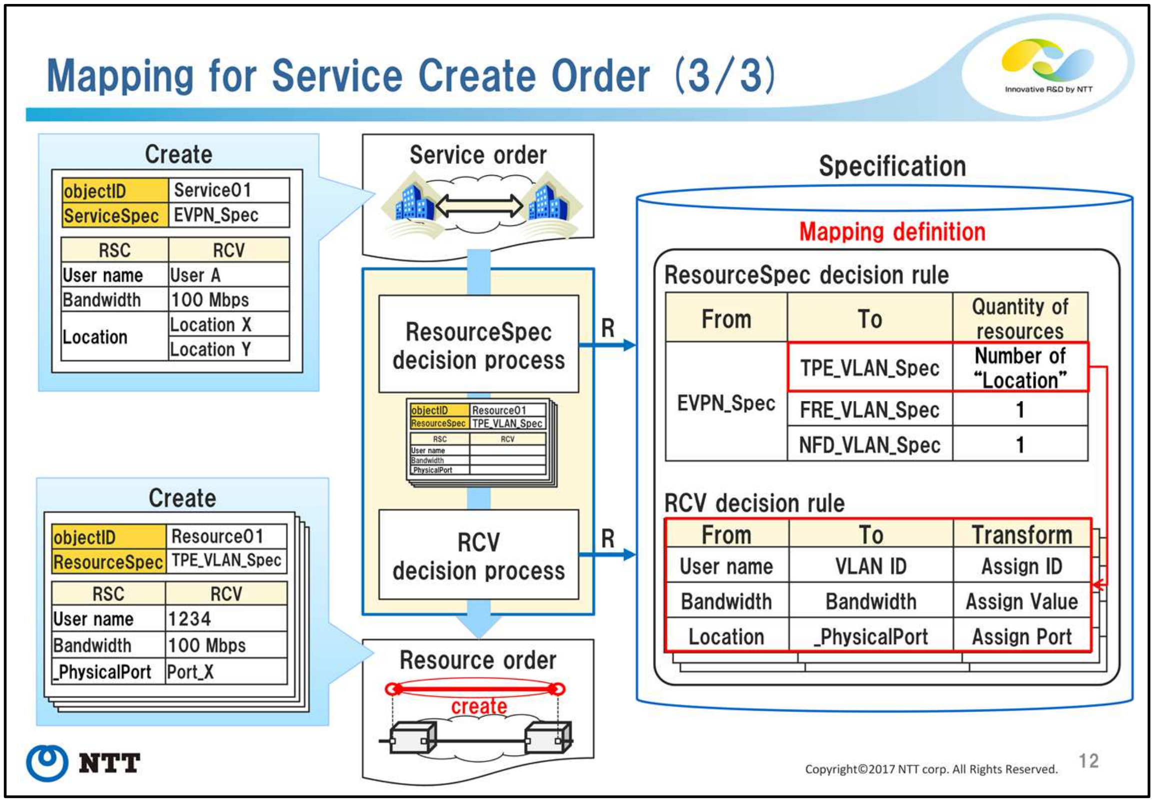


First, we explain the ResourceSpec decision process in detail.

The ResourceSpec decision process determines the specification and number of resources necessary for providing a service based on the ResouceSpec decision rule defined in the mapping definition. In the ResourceSpec decision rule, information of "From", "To", and "Quantity of resources" is described. The object of "From" is a service specification, the object of "To" is a resource specification, and "Quantity of resources" is information for determining how many resources of a given specification should be created. In "Quantity of resources", in addition to natural numbers, rules, such as number of "Locations", are described. The ResourceSpec decision process first obtains a ResourceSpec decision rule of which the "From" ServiceSpec is in accordance with the service specification contained in the service order. Next, resources based on the "To" resource specifications are created according to "Quantity of resources". At this time, the object identifier, or ID, for uniquely identifying the resource is determined.

This figure shows an example of the proposed order-mapping function upon receiving the service-create order for an EVPN, which refers to "EVPN_Spec". The ResourceSpec decision process acquires a ResourceSpec decision rule with EVPN_Spec as the "From" and creates resources referring to the resource specification of each "To". As a result, two resources that refer to the TPE_VLAN_Spec, one resource referring to the FRE_VLAN_Spec, and one resource referring to the NFD_VLAN_Spec are created.

At this point, the created resources have no RCV for the attributes defined in the RSC.
Next, we show the RCV decision process in detail.



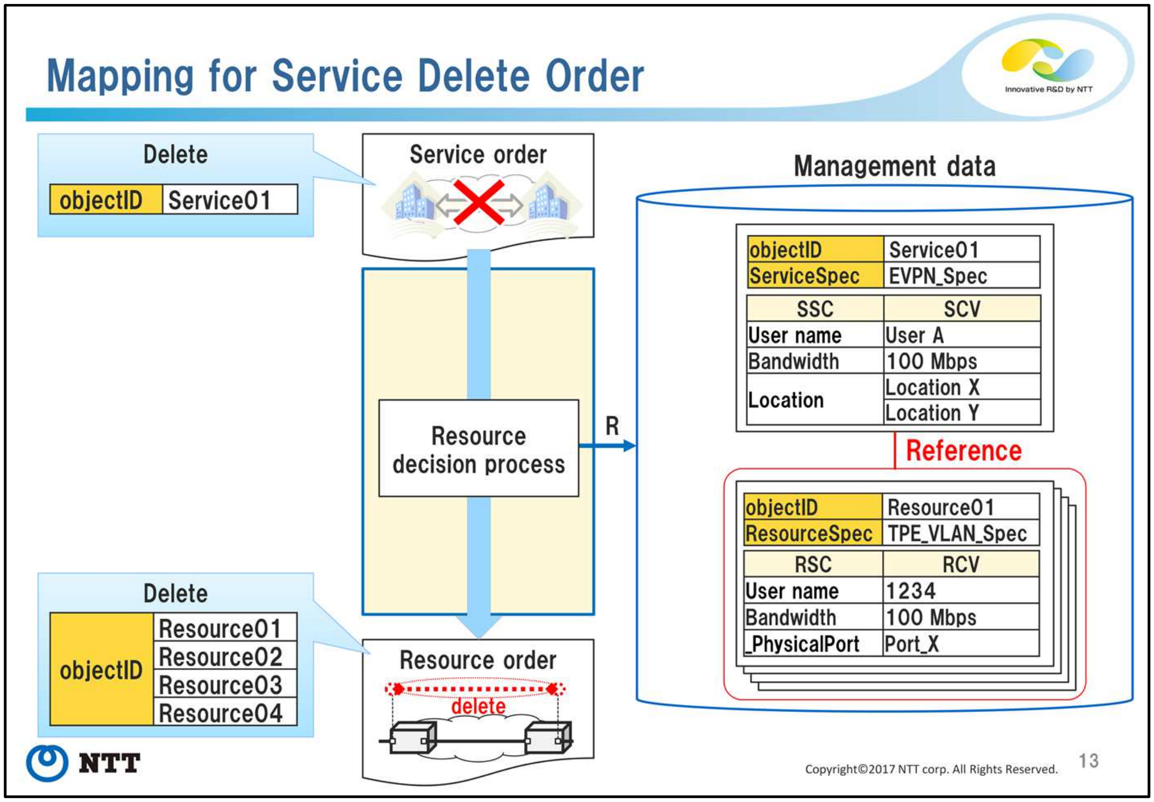
The RCV decision process is a process to determine the RCV of each resource created by the ResourceSpec decision process. The RCV decision process is based on the RCV decision rule included in the proposed order mapping definition. One RCV decision rule is created for each "To" in the ResourceSpec decision rule.

In the RCV decision rule, information about "From", "To", and "Transform" is described. The SSC for "From", RSC for "To", and value conversion rules for "Transform" are described. For example, "Assign ID", "Assign Value", "Assign Port", and the like are conceivable as the rule specified as "Transform". "Assign ID" is a process of searching and determining the ID using the SCV of the SSC specified by "From" as a key, "Assign Value" is a process of substituting the SCV of the SSC designated by "From", "Assign Port" involves assigning the same location to the physical port as the SCV of the SSC specified by "From". The RCV decision process converts the SCV of the SSC described in "To" according to the conversion rule of "Transform" and determines the RCV of the RSC described in "To".

For the resource referencing the TPE_VLAN_Spec in the figure, we give an example in which the RCV is determined based on the RCV decision rule.

"Assign ID":
search and determine the value of the VLAN ID of the resource from the value of the user name of the service,
"Assign Value":
substitute the value of the bandwidth of the resource from that of the service,
"Assign Port:":
assign the value of the connection location of the service from the value of the connection location of the service.

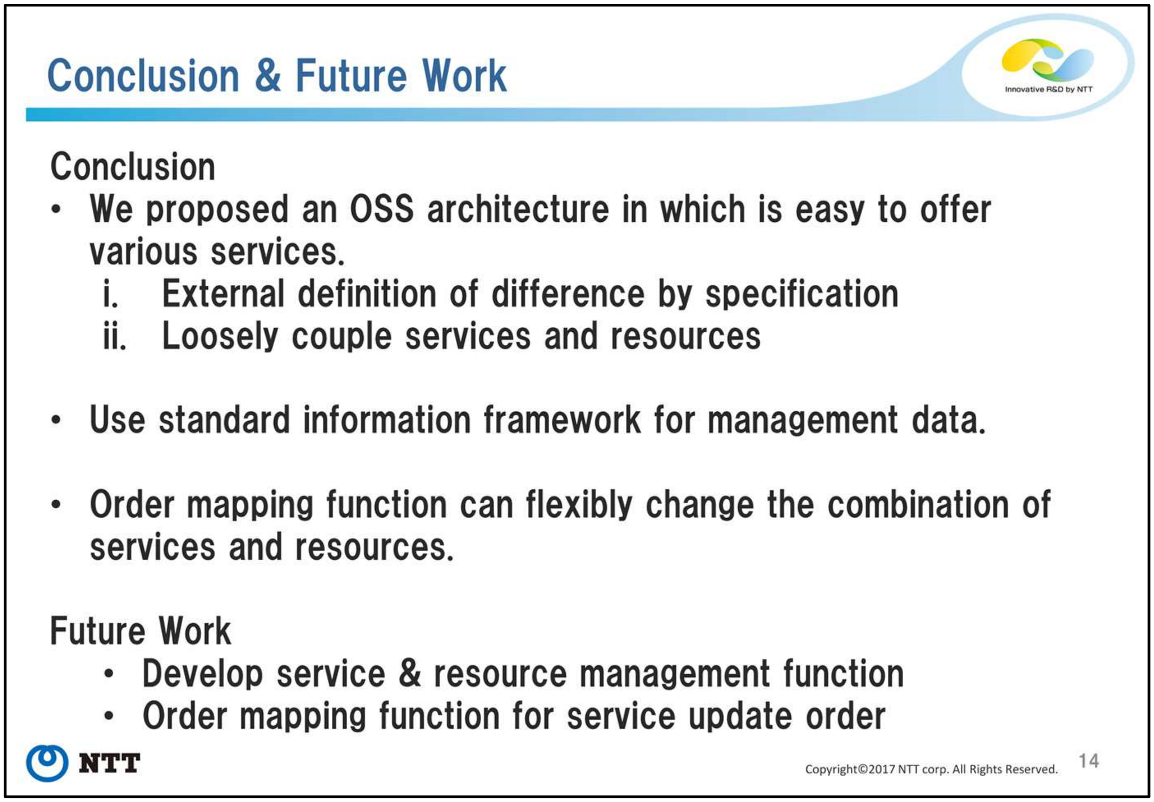
The proposed order-mapping function sends a command to create resources determined by the above process to the resource-management function. This completes the mapping of the service-create order.



5.2.2 Mapping for Service Delete Order

The proposed order-mapping function for a delete order translates a service-delete order to a resource-delete order, which requests the deletion of resources providing the service. It is assumed that the service-delete order describes the "objectID" of the service to be deleted.

The proposed order-mapping function consists of only the resource decision process, which determines deletion targets. Deletion targets are resources used by the service to be deleted, in other words, resources created by the translation of a create order for the service. The resource-decision process determines the targets by using references of service and those resources. The proposed order-mapping function sends a resource-delete order that requests deletion of the resources to the resource-management function..



6. Conclusion and Future Work

For B2B2X business, we discussed our proposed OSS architecture to make it easy to offer various services. Our approaches are external definition of the difference for each service type and loose coupling of services and resources. Compared with a legacy architecture, our proposed architecture can suppress an increase in development cost due to an increase in service types. Also, we studied specifications and management data and proposed an order-mapping function to develop an OSS based on our architecture. We consider that the SID, which defines management models, is useful for specifications and management data in our architecture. Our order-mapping function translates orders for creating or deleting a service into orders for resources according to the mapping definition for each specification.

For future work, we will study in detail management functions that operate according to specifications. To be able to update parameters of a service, we will develop an order-mapping function that translates service-update orders to resource orders.

References

[1] TM Forum, https://www.tmforum.org.

[2] TeleManagement Forum GB921, “Core Frameworks Concepts and Principles: Business Process, Information and Application Frameworks,” Version 16.0.1, April 2016.

[3] TeleManagement Forum GB922, “Information Framework (SID): Service Overview Business Entities,” Version 16.0.1, May 2016.

[4] TeleManagement Forum GB922, “Information Framework (SID): Physical Resource Business Entities,” Version 16.0.1, May 2016.

[5] TeleManagement Forum GB922, “Information Framework (SID): LogicalResource and ConmpoundResource Bisiness Entities,” Version 16.0.2, May 2016.