Configuration Method for Integrated Interface Rule

Jae-Hyoung Lee† and Seung-Ik Lee††, Shin-Gak Kang†††

†Broadband network engineering, University of Science and Technology, Gajeong-ro, Yuseong-gu, Daejeon, 305-350, Korea
††Standards Research Center, ETRI, 218 Gajeong-ro, Yuseong-gu, Daejeon, 305-700, Korea
†††Standards Research Center, ETRI, 218 Gajeong-ro, Yuseong-gu, Daejeon, 305-700, Korea

Summary
Until now, the web service process system was carried out in a static environment. However, the external processes or services specified at design time, rather than a static environment it has been for technical studies on run-time inputs to configure the service to be able to add new services, applied service for the techniques of dynamic Service binding.

There are multiple service instances for the service type of one. However, since the structure of interfaces is independent so that is difficult to interaction. Even if interfaces perform the same function, function structure of each interface is independent that it has less efficiency as a result internal unnecessary procedures and hard interaction. The function of internal interface has the same mechanism, but Services are constrained to vary the structure of the codes.

In this paper, when a user requests a service, service is treated by applying the abstracted specifications to be defined the structure of each interface. The service response time, we propose a method that provides the user with a result of processing by inverse transform according to the structure of each interface specification.

To provide the optimum service does not depend on the characteristics of each interface by configuring the abstracted specification.

Key words:
Dynamic service binding, abstracted spec, BPEL, interface translation,

1. Introduction
In present, SOA [1] was created a technique for linking and integrating information from distributed environment based on the Web. Variety of technologies and services related to SOA, such as BPEL, Open API, and Mashup. As a means for defining workflow between services represented by a single business process by combining the various services, BPEL [2] (Business Process Execution Language) is used. Business Process Execution Language (BPEL) is a standard language used to define and execute business processes in a Web services environment. By using BPEL, you may want to implement intuitive and easy composite service called “business processes” in Web services. In addition to BPEL, anyone can possible to develop applications and services by opening the Open API from public institutions, Google and Naver. Mashup [3] became possible to produce a new service form that existing contents and services via Open API(Application Programming Interface). Mashup is used to encompassing songs which consist entirely of parts of other songs in the musical genre. Representative Companies of web service such as Google and Amazon, Naver are opening their services to support a variety of application development.

The each company or individual have in their system from outside the Open API to integrate and to provide new services. For example, using Google Open API provides to the optimal path by mapping public transportation of surrounding area or destination path through the map information.

Until now, the web service process system was carried out in a static environment. However, the external processes or services specified at design time, rather than a static environment it has been for technical studies on run-time inputs to configure the service to be able to add new services, applied service for the techniques of dynamic Service binding.[4]

In order to provide optimal services to the user, various methods have been studied. The user request services to system that find the appropriate component configure and provide services. At this time e different structure of each interface it is difficult to provide services optimized. The functions of internal interface have the same mechanism, but Services are constrained to vary the structure of the codes. In this paper, when a user requests a service, service is treated by applying the abstracted specifications to be defined the structure of each interface. The service response time, we propose a method that provides the user with a result of processing by inverse transform according to the structure of each interface specification. To provide the optimum service does not depend on the characteristics
of each interface by configuring the abstracted specification.
In Chapter 2, we analyze the problem of functionality in
the present, in the chapter 3, we introduce the main
research on dynamic service existing binding, in the
chapter 4, proposed to applying the concepts and scenarios
interface translation, in the chapter 5, explain configured
source code by the WSDL and BPEL [5] for the proposed
method, in chapter 6, we present the research methods of
the future and conclusion.

2. Problem Statement

In general, services and processes are using the technique
of static service binding that reference to define the
external services at design time. However, scale of the
process are increased and complicated as interacting
between single services and adding the other services. In
addition, whenever a change is made to an external service,
occurs changing the entire process. To solve this problem,
we are used to the technique of dynamic service binding.
While service is provided to user, service disconnected
between user and provider or cannot provide a service by
network environment, server or device failure, malfunction.
If the interface is not able to provide the service due to a
problem on the device or network, it should be replaced
with another interface.
In other words, if one of the components cannot provide
properly service, its affected to the rest of the service can't
provide a seamless service. Dynamic service binding is
available run-time binding between processes, and
eliminate to set up and manage at the design time. The
redundant process can be reduced between parent and
child processes in case of changes or modifications and the
parent process, can be accessed automatically sub-process
of changes and improvements.
There are multiple service instances for the service type of
one. However, since the structure of each interfaces are
independent so that is difficult to interaction. Even if
interfaces perform the same function, function structure of
each interface is independent that it has less efficiency as a
result internal unnecessary procedures and hard interaction.

3. Related Work

Service-oriented architecture (SOA) is a software design
and computer architecture design pattern based on
structured collections of discrete software modules, known
as services that collectively provide the complete
functionality of a large software application. Competitive
companies are emphasis on the collaborative relationship
at business environment to the rapidly changing business
environment.
In order to, it is essential to avoid disruption of business
processes in place to enable the real-time information
sharing, inter-organizational, inter-company, gather real-
time data access. Web service and SOA can
interoperability between the different business processes
and applications, and can be defined through the BPEL.
Based on the service model, BPEL(Business Process
Execution Language) is a standard language used to define
and execute business processes in a Web service
environment. BPEL [6] is used to WSDL that express the
information and meaning of web service by an XML-based
language, mostly large companies such as Microsoft, IBM,
Oracle, are leading and using the promotion.
Generally, the service process is consists of a static binding
and dynamic binding. At the design time, static binding
mechanism is defined the logic and service type for the
external process. Ways in which services can be managed
actively or add a new external service is called dynamic
binding.
In the past, as static service techniques, BPEL have
disadvantages for workflow management and service
selection in the design to be done manually but recently
BPEL is defined the dynamic binding mechanism to
support interoperability with the external service
partner links.
In other paper, they show that interface translation [7],
when the service invoke and reply to complex services
through the service binding control function based on the
binding information. However, it did not mention concepts
and reason but mention the translation process.

4. Proposed Scheme

BPEL is used in a static method. However, we propose on
BPEL workflows by applying dynamic service binding at
time to provide proactive service.
Therefore, it has the effect of reducing the need to predict
and manage existing design time. In addition, the interface
structure is different between the service instance,
depending on the type of service that is required when
calling a Web service receives a request for a service from
the user, when the system changes is difficult to modify the
process. To solve this problem, the aims to clear and
shows the configuration for the interface translation
process to convert the structure of the interface
specification of abstract integration, by applying the
scenario. In this paper, we apply for the transportation
system to aid in the understanding of the scenario. For
example, the customer supposed to delivery requests for
goods transportation service system. Transportation
service get the basic information regarding the recipient's
address and the sender address from the customer, then service processes will determine the appropriate delivery company. In this case, the syntax or structure used by the each of delivery companies must be pre-defined at design time, if you apply a dynamic service binding mechanism, flexible service that can provide additional advantages at runtime.

In this paper, we propose how to place created by the abstracted specification rule of the structure of the delivery service, the structure of each interface be translated by applying the rules of the abstracted specification when the call to service, when the service response, the structure of abstracted specification rule be reverse translated according to the structure of the interface and processing services.

The figure 1 shows the change of concept for the proposed method and the existing service. When a user requests a service, handling and processing services are depending on structure of each interface to determine. The appropriate interface according to the process that the response back to the user in the existing method was used. The way that we propose, when user requests a service, the process is to find appropriate interface. Even they have different structures of each interface but the basic functions and services of the same type, so that interface can be replaced to another interface, if you are having trouble with one interface. Once you have determined the appropriate interface, service translate and process by mapping the abstract spec rule, a result of process gives by reverse translation to existing interface structure.

To this, the process of applying the scenario for transportation service implement BPEL and will be described the presenting simple rules to propose

Translation in WSDL files and papers about transportation companies such as external interface.

The following example file shows the implementation of BPEL [8] type about transportation service process.

```
<process name="TransportationService"
  TargetNamespace="http://jbpm.org/examples/bpel/delivery/">
  xmlns:wsa="http://schemas.xmlsoap.org/ws/2003/03/addressing"
  xmlns:tns="http://jbpm.org/examples/service/delivery/"
  xmlns=http://schemas.xmlsoap.org/ws/2003/05/partner-link/DeliveryService
  <partnerLinks>
    <partnerLink name="DeliveryService"
      partnerLinkType="services:DeliveryService"
      myRole="DeliveryServiceRequester"
      partnerRole="DeliveryServiceProvider"/>
    <partnerLink name="DeliveryService"
      partnerLinkType="services:DeliveryService"
      myRole="DeliveryServiceRequester"
      partnerRole="DeliveryServiceProvider"/>
    <partnerLink name="client"
      partnerLinkType="trp:transportationLT"
      myRole="transportationService"
      partnerRole="transportationServiceCustomer"/>
    <partnerLink name="ProductTransportationStatus"
      partnerLinkType="prd:productLT"
      partnerRole="ProductTransportationStatusService"/>
    <partnerLink name="EaglesDelivery"
      partnerLinkType="del:DeliveryLT"
      myRole="DeliveryCustomer"
      partnerRole="DeliveryService"/>
    <partnerLink name="LionsDelivery"
      partnerLinkType="del:DeliveryLT"
      myRole="Delivery Customer"
      partnerRole="Delivery Service"/>
    <partnerLink name="Giants Delivery"
      partnerLinkType="del:DeliverLT"
      myRole="DeliveryCustomer"
      partnerRole="DeliveryService"/>
  </partnerLinks>
  <partnerLinkBinding name="DeliveryService">
    <property name="wsdlLocation">DeliveryService.wsdl</property>
  </partnerLinkBinding>
  <variables>
    <variable name="DeliveryInput" messageType="DeliveryInputMessage"/>
    <variable name="DeliveryOutput" messageType="DeliveryOutputMessage"/>
  </variables>
  <sequence>
```

Figure 1 : Changes in the service concept for the proposed method and the conventional method
<invoke partnerLink="DeliveryService"
portType="DeliveryAvailibilityPT"
operation="DeliveryAvailibility"
inputVariable="DeliveryDetails" />

<receive partnerLink="DeliveryService"
portType="DeliveryCallbackPT"
operation="DeliveryReceiptCallback"
variable="DeliveryOutput" />
</sequence>
</process>

Delivery provider for the content of the registered service
as DeliveryService.wsdl file, the example below is
expressed in WSDL. [9]
<definitions targetNamespace =
"http://jbpm.org/examples/bpel/delivery/">
<service name="EaglesDelivery">
<portname="DeliveryServicePort"
binding="tns:DeliveryServiceBinding">
<soap:address location =
"http://localhost:9700/orabpel/default/EaglesDelivery"/>
</port>
</service>

<service name="LionsDelivery">
<portname="DeliveryServicePort"
binding="tns:DeliveryServiceBinding">
<soap:address location =
"http://localhost:9700/orabpel/default/LionsDelivery"/>
</port>
</service>

<service name="GiantsDelivery">
<portname="DeliveryServicePort"
binding="tns:DeliveryServiceBinding">
<soap:address location =
"http://localhost:9700/orabpel/default/GiantsDelivery"/>
</port>
</service>

Instead of defining the delivery service of the individual, I
was using the type and delivery service name generic. You
can be sure that each delivery provider is registered in the
<service>.

5. The proposed specification for interface
translation

Each transportation type of service and service instance is
written in the WSDL interface. Suppose as follows by
applying the above scenario, the basic structure for the
sender and the recipient of the shipping service, each has a
different structure.

The translation rules that we propose can be defined as the
functional structure for the address of the sender and the
recipient of their respective companies.

![Diagram](image)

Figure 2: Example scenario to convert the interface structure of each
delivery company to abstracted specification.

- Translation example by pseudo code

  get send address from interface A
  save space a <- send address
  get source address from spec rule
  save space b <- source address
  call swap with space a and space b
  return to process

After applying the rule, if the process determines the
services, the reverse conversion work to fit the original
structure of the interface to be applied to the selected
interface is translation work to the contrary

- reverse translation example by pseudo code

  get source address from spec rule
  save space a <- source address
  get send address from interface A
  save space b <- send address
  call swap with space a and space b
  return to process.

6. Conclusion

Summary and expected effect of the proposed schema In
this paper, single abstracted spec rules by providing to
unify the interface between structure and to provide a more seamless service for in order to solve the problems that arise due to the different structure and the same features of the interface, select one services provided at. Existing, structure of interfaces must have the same in the design. But, the proposed method can be applied at runtime and be also possible to replace the interface between them when the problem occurs. The proposed method describes the basic concept of interface translation, and showed that the simple logic through the simple pseudo code. In the future, we will require the implementation of the programming language and verification through simulation. Based on the simple scenario of a virtual explained in easy to understand. But, which can be applied in complex services or services will require more research. Studies can be applied in situations or services complex is required.

References
[9] http://blog.naver.com/PostView.nhn?blogId=sarangbjh&log No=80098930760&redirect=Dlog&widgeTypeCall=true

Jae-Hyoung Lee received the B.E. degree from Kongju Univ. in 2010, he stayed in Standard Research Center (SRC), ETRI(Electronics and Telecommunications Research Institute). He is M.S.Student of UST(University of Science & Technology).

SEUNG-IK LEE received his B.S. degree in computer science and engineering from Handong University, Korea, in 2000, and his M.S. and Ph.D. degrees in computer science from Korea Advanced Institute of Science and Technology (KAIST) in 2002 and 2009, respectively. He joined the Standards Research Center of the Electronics and Telecommunications Research Institute (ETRI), Korea, in May 2009, and has participated in the standardizations of IPTV, multicast, and NGSON in ITU-T SG13, JTC1/SC6, and IEEE P1903. His research interests include IPTV, mobile multimedia transport, and future networks.

SHIN-GAK KANG received B.S., M.S., and Ph.D. degrees in electronics engineering from Chungnam National University, Daejeon, Korea in 1984, 1987, and 1998, respectively. He joined ETRI in 1984 and is currently director of the Smart Communication Standards Research Team in the Standards Research Center. He has actively participated in many international standards bodies such as ITU-T SG11, SG13, SG16, JTC1/SC6, and IEEE NGSON as Rapporteur, Convenor, and Editor. His current research interests include multicast, IPTV/smart TV, and service overlay and content networking.