1-1-2010

Managing conflict of interest in service composition

Haiyang Sun
Macquarie University

Weiliang Zhao
University of Wollongong, wzhao@uow.edu.au

Jian Yang
General Research Institute for Non Ferrous Metals,Ministry of Science & Technology, China,Macquarie University

Managing Conflict of Interest in Service Composition

Haiyang Sun, Weiliang Zhao, and Jian Yang

Department of Computing, Macquarie University,
Sydney, NSW2109, Australia
{hsun,wzhao,jian}@ics.mq.edu.au

Abstract. Web services can be composed of other services in a highly dynamic manner. The existing role based authorization approaches have not adequately taken component services into account when managing access control for composite services. In this paper, we propose a service oriented conceptual model as an extension of role based access control that can facilitate the administration and management of access for service consumers as well as component services in composite web services. Various types of conflict of interest are identified due to the complicated relationships among service consumers and component services. A set of authorization rules are developed to prevent the conflict of interest. This research is a step forward to addressing the challenge in authorization in the context of composite web services.

Keywords: Authorization, Conflict of Interest, Composite Web Services.

1 Introduction

The nature of web service creates the opportunity for building composite services by combining existing elementary or complex services (referred to as component services) [1]. Authorization of composite web services is different from traditional authorization in a close system due to the dynamic and complex relationships among service consumers and component services. Let us look at an example of Tom & Brothers which is a vehicle parts dealer that provides vehicle engines and engine accessories for both military and civil use. An *Order Service* is set up in Tom & Brothers including five operations: (1) *Order Engine*, (2) *Order Engine Accessory*, (3) *Payment*, (4) *Payment Verification* and (5) *Logistics* (See Fig. 1). Note, the *Logistics* operation is not available to the military customers since they organize parts shipment by themselves. When receiving a part order from a customer, the Tom & Brothers will order the parts from various parts suppliers. As soon as the payment has been verified, the goods will be transported to the customer. We observe that the following features exist in *Order Service* in Tom & Brothers that make authorization of composite web services complicated:

– **Complicated Authorization Constraints:** The component services of a composite web service may belong to different organizations, come from

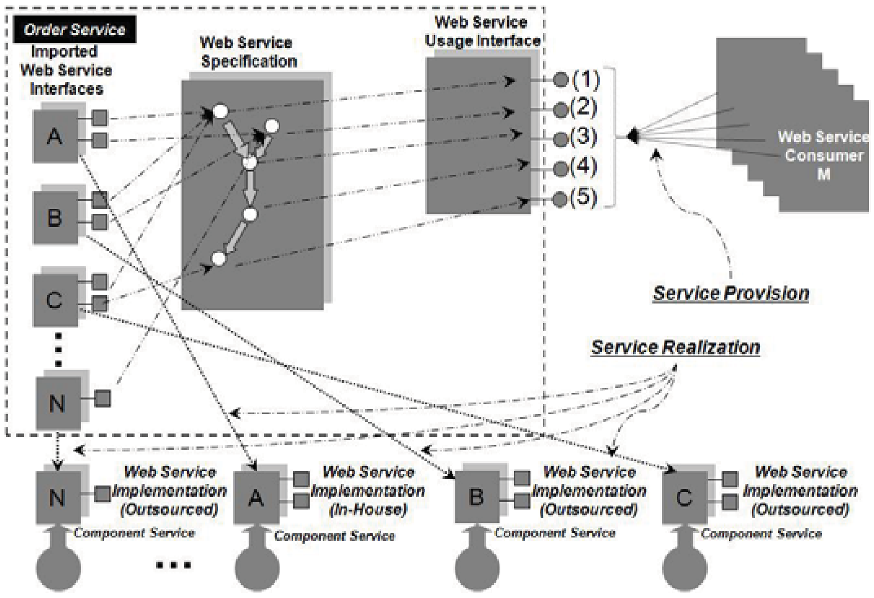
Order Service
Imported Web Service Interfaces
Web Service Specification
Web Service Usage Interface
(1)
(2)
(3)
(4)
(5)
Web Service Consumer M
Service Provision
Service Realization
A
B
C
N
Web Service Implementation (Outsourced)
Web Service Implementation (In-House)
Component Service

Fig. 1. *Order Service* in Tom & Brothers

different security domains, and have different security and interest requirements. The authorization constraints of a composite web service can be very complicated. For example, in Fig. 1, the operation (1)-*Order Engine* can be supported by the component web services A, B and C. Therefore before authorization is granted to a service consumer for the *Order Engine* operation, the policies of component web services A, B and C need to be checked. The complicated authorization constrains of composite services change the basic authorization question ”`who can do what?`” to a more complicated one as ”`who can do what under what conditions`”.

– **Dynamicity of Component Services:** There may be several web services that can provide the same or similar operations. The specifications and policies of individual component services can change frequently. For example, if a component service changes its authorization policy from asking Tom & Brothers for professional engineer certificate to requiring sales representative qualification, then all the service operations in Tom & Brothers that are supported by the component service need to update their authorization policies accordingly. In Fig. 1, component services A, B, and N can support the same type of engine accessories to Tom & Brothers. If the changes occur frequently or happen in many web services, an efficient way to manage these changes is needed.
– **Conflict of Interest:** Authorization in composite services must prevent conflicts of interest among service consumers, among component services, and between consumers and component services. When there is a conflict of interest between a specific service consumer and a specific component service, the *Order Service* in Tom & Brothers should not be authorized to the service consumer when this component service is essentially needed in the composite service. For example, USA military customers may have a conflict

of interest with a component web service from a Chinese part supplier. The existing role based access control employs the mechanism of separation of duty to deal with conflict of interest for consumers, which is inadequate in dealing with the complicated situations occurred in the service setting.

In Role Based Access Control (RBAC) [2], users acquire permissions through their roles rather than that they are assigned permissions directly. This greatly reduces the administrative overhead associated with individual users and permissions. All existing role-based models in web service paradigm have not brought the administration of component services into the picture. The component services are normally remote resources or related with remote resources (the term resource will be used instead of component service later in the paper). The quantity of resources can be very large and they can be prone-to-change, which must be considered in web service authorization. In research work [8, 10, 11], roles are assigned to service consumers for service authorization. However all these researches have not put resources into the picture or they simply employ an unrealistic assumption that there is a *global* coordination on internal authorization policies of each autonomous web services to enforce the access control in service composition. Furthermore, resources in composite web services can introduce new types of conflict of interest on top of the conflict of interest between service consumers defined in the traditional RBAC approaches. The conflict of interest can occur among service consumers, among resources, and between service consumers and resources.

In this paper we propose a general approach for the authorization of composite web services as an extension of role based access control, which grants authorization to a service consumer based on the authorization constraints of the composite web services as well as those of the resources. Based on the previously proposed Service Oriented Authorization Control (SOAC) in [4], four types of conflict of interest are identified regarding to both service consumers and resources invoked in composite web services. The authorization rules are devised for these identified types of conflict of interest. Comparing with existing work, our proposed approach has the following merits:

- The characteristics and requirements for both service consumers and resources in composite web services can be explicitly captured.
- The proposed approach provides an efficient way to administrate and manage large number of service consumers and dynamic resources in relation to authorization in composite web services.
- The proposed approach has the capability to detect the conflict of interest among service consumers, among resources, and between service consumers and resources that are far more complicated than the ones identified in the existing role based authorization approaches for web services.

The rest of paper is organized as follows. Section 2 describes the conceptual model for the service oriented authorization. Section 3 identifies the various types of conflict of interest and provides the rules to prevent the conflict of interest in authorization of composite web services. Section 4 overviews some related work. Concluding remarks and discussion of future work are presented in Section 5.

2 Conceptual Model of Service Oriented Authorization Control

In this section, we describe a conceptual model, named as *Service Oriented Authorization Control (SOAC)* for managing the authorization of composite web service. SOAC is divided into two parts, *service provision* and *service realization* (See Fig. 2). We express the SOAC conceptual model by using the notation of **Entity-Relationship** (**E-R**) **Diagram**. In Fig. 2, rectangles represent elements and diamonds represent relationships.

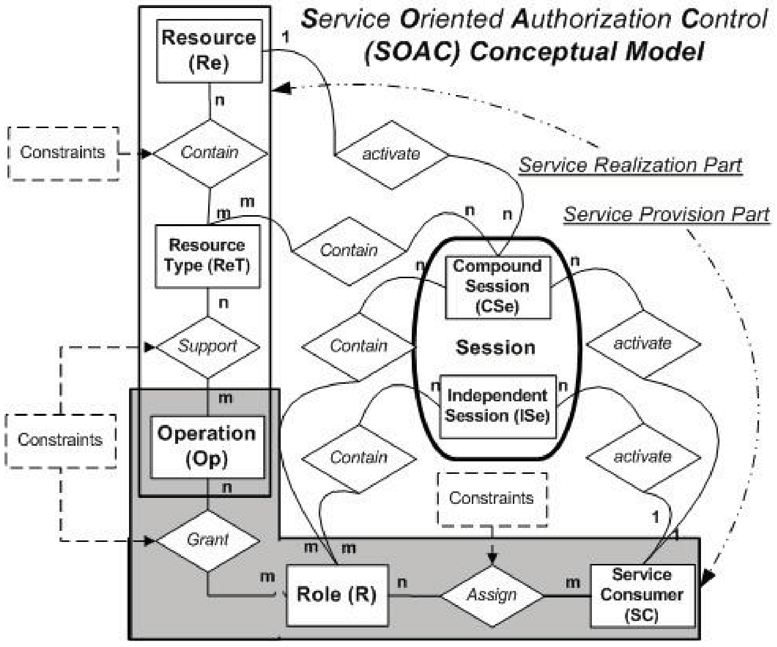

Fig. 2. Service Oriented Authorization Control (SOAC) Conceptual Model

2.1 *Service Provision* Specification

In *service provision*, a service consumer can get the authorization by fulfilling constraints of the composite service (See *Constraint* enacted between the elements of Role (**R**) and Service Consumer (**SC**) in Fig. 2). In Fig. 2, we define service consumer as the element that requires to access the composite web service's operations (**Op**). Since service consumers are prone to change and the quantity of consumers can be vary large, directly specifying the assignment of operations to individual service consumers needs tedious administration efforts. In SOAC, we follow the philosophy of RBAC to have concept role to encapsulate the service consumers that can satisfy the common authorization constraints of composite web services. A role will be assigned to a service consumer based on its characteristics (typically a credential that service consumer submits to the composite web service). Each role binds with a group of operations that can be accessed. The roles guarantee that the composite web service's operations can only be accessed by the qualified service consumers. The mapping between service consumers and roles are considered in the *service provision* part of SOAC with the following formal specification.

Definition 1. *The service provision in SOAC includes:*

- ***SC**, **R** ,and **Op** are elements representing Service Consumer, Role, and Operation.*
- **SCA** $\subseteq$ ***SC*** $\times$ ***R***, *a many-to-many relation to map service consumer to role assignment. Formally,* $\forall s^c \in SC$, $\forall r \in R$, $(s^c, r) \in SCA \Rightarrow s^c.credential = r.credential$, *where the credential that the service consumer submits is consistent with the credential that the role requires.*
- ***assigned_sc:(**r**:R)*** $\rightarrow 2^{SC}$, *the mapping of role r onto a set of service consumers. Formally,* $assigned\_sc(r) = \{s^c \in SC | (s^c, r) \in SCA\}$.
- **OPA** $\subseteq$***Op***$\times$***R***, *a many-to-many relation to map operation to role assignment.*
- ***assigned_op:(**r**:R)*** $\rightarrow 2^{Op}$, *the mapping of role r onto a set of operations. Formally,* $assigned\_op(r) = \{op \in Op \mid (op, r) \in OPA\}$.

2.2 *Service Realization* Specification

Due to the feature of **Dynamicity** of resources, it is unrealistic to specify the relationships between resources and the supported operations of composite web services individually. Resource type is defined for a set of resources by identifying their characteristics and authorization constraints (See Fig.2). The composite web service can bear multiple resource types that cover many resources. The resources can be accessed to support the operation if the operation is mapped with a resource type that covers these resources. Resources are linked with resource types with constraints. (See *Constraint* between the elements of Resource Type (**ReT**) and Resources (**Re**) in Fig. 2). The mapping between resources and resource types is the major concern in *service realization* part of SOAC. The formal specification of *service realization* is presented here.

Definition 2. *The service realization in SOAC includes:*

- ***Op**, **ReT**, and **Re** are elements representing Operation, Resource Type, and Resource.*
- **SPA** $\subseteq$ ***Op*** $\times$ ***ReT***, *a many-to-many relation to map operation to resource type.*
- ***assigned_ret:(**ret**:ReT)*** $\rightarrow 2^{Op}$, *the mapping of resource type ret onto a set of operations. Formally,* $assigned\_ret(ret) = \{op \in Op | (op, ret) \in SPA\}$.
- **RTA** $\subseteq$***Re***$\times$***ReT***, *a many-to-many relation to map resource to resource type. Formally,* $\forall re \in Re$, $\forall ret \in ReT$, $(re, ret) \in RTA \Rightarrow re.constraint = ret.constraint$, *where the constraint that restricts the access on the resource is consistent with the constraint that the resource type can fulfill.*

– ***assigned_re:(**ret**:ReT)*** $\rightarrow 2^{Re}$*, the mapping of resource type ret onto a set of resources. Formally, assigned_re(ret)=*$\{re \in Re | (re, ret) \in RTA\}$.

2.3 Integration of Service Provision and Service Realization

Service provision and *service realization* in SOAC must be worked together for authorization of composite web services. In Fig. 2, the mappings between the elements of Role (**R**), Operation (**Op**), and Recourse Type (**ReT**) integrate the *service provision* and *service realization.* The access to the composite web service can be assigned to a service consumer if all the constraints of the composite web service and its resources can be satisfied. In *service provision*, the service consumer is assigned a specific role for the access to the operations; while in *service realization*, the operations are mapped with resource types that cover all resources required.

In order to check conflict of interest at runtime, element **Session** is introduced at integration of *service provision* and *service realization* in SOAC (see Fig. 2). There are two types of sessions, *Independent session* (**ISe**) and *Compound session* (**CSe**). **ISe** is used to check runtime conflict of interest in *service provision*; while **CSe** is used to check runtime conflict of interest at integration of *service provision* and *service realization.* After a service consumer starts to send message to the composite web service for accessing its operations, the service consumer activates the assigned specific roles in an independent session. The resource types are involved when resources are required in composite web services. A compound session is established when the message from service consumer is transferred to resource. In this case, specific resource type is activated by the resource as well as the role is activated by the service consumer in a compound session. Note, the resource type and associated resources can not be included in the independent session, since a composite web service can not use the resource type without receiving the authorization request from the service consumer. Below is the formal definition of **Session**.

Definition 3. ***Session*** *includes two types, Independent Session (**ISe**) and Compound Session (**CSe**).*

– ***Independent Session (ISe)*** *is used by service consumer* s^c *to map the set of activated roles* $\{r_1..r_j\}$*,* $j \geq 1$*.*
– ***Compound Session (CSe)*** *is used by a pair of service consumer and resource* $< s^c, re >$ *to map a set of activated roles and resource types* $\{< r_1, ret_1 >..< r_j, ret_k >\}$*, (Note, the operations that the service consumer* s^c *requires to access are the same operations that the resource re can provide support to.) where:*
  ◦ $r_1..r_j$*,* $j \geq 1$*, is a subset of roles assigned to and activated by the specific service consumer* s^c*.*
  ◦ $ret_1..ret_k$*,* $k \geq 1$ *is a subset of resource type assigned and activated by the specific resource.*
– ***Service Consumer Independent Session:*** *SCSi:(*s^c***:SC***)*$\rightarrow 2^{ISe}$*, the mapping of service consumer* s^c *onto a set of independent sessions ISe.*

- ***Service Consumer Compound Session:*** *$SCSc : (s^c : SC) \rightarrow 2^{CSe}$, the mapping of service consumer s^c onto a set of compound sessions CSe.*
- ***Role Independent Session: RSi:***$(se_i$***:ISe)***$\rightarrow 2^R$*, the mapping of independent session se_i onto a set of roles.*
- ***Role Compound Session****: $RSc : (se_c : CSe) \rightarrow 2^R$, the mapping of compound session se_c onto a set of roles.*
- ***Resource Session****: RES(re:Re)$\rightarrow 2^{CSe}$, the mapping of resource re onto a set of compound session CSe.*
- ***Resource Type Session****:RTS(se_c:CSe) $\rightarrow 2^{ReT}$, the mapping of compound session se_c onto a set of resource types.*

3 Management of Conflict of Interest

Four types of **Conflicts of Interest** are identified based on SOAC. Authorization rules are defined to prevent the various types of conflict of interest at both design time and run time.

The relationships between two elements with the same type in SOAC are defined as *Exclusive* $\otimes$ or *Non-exclusive* $\ominus$. *Exclusive* relationship means that two elements of SOAC, e.g., two service consumers, two roles, or two operations, are ostracized each other; while *Non-exclusive* relationship means that two elements of SOAC are not ostracized each other. The relationship between elements with the same type in different authorizations should be the same; Otherwise, conflict of interest will occur.

3.1 Conflict of Interest between Service Consumers

In *service provision*, the relationship between two service consumers should be the same as the relationship between the assigned roles for these two consumers to prevent conflict of interest. In Fig. 3, if Op_a and Op_b are *exclusive* ($Op_a \otimes Op_b$), then the relationship between R_i and R_j that are mapped to the operations Op_a and Op_b respectively should reflect the *exclusive* relationship ($Op_a \otimes Op_b \Rightarrow R_i \otimes R_j$). The relationship between assigned roles for service consumers SC_n and SC_m must be matched with the relationship between these two consumers. If service consumers SC_n and SC_m are *non-exclusive* with each other ($SC_n \ominus SC_m$), then SC_n and SC_m can not be assigned roles R_i and R_j respectively at the same time because the roles have the *exclusive* relationship.

Two special cases are illustrated in Fig. 3, where (1) two service consumers become the same one in special case A, and (2) two operations become the same one in special case B. Moreover, the relationship between the element and itself can be *non-exclusive* or *exclusive* according to its situation.

For example, *Payment* and *Payment Verification* are *exclusive* operations that need to be mapped to different roles, and such roles are recognized as *exclusive* roles as `Payer` and `Verifier`. If a service consumer is assigned with both `Payer` and `Verifier` (Special case A in Fig.3), the conflict of interest will occur, since `Payer` and `Verifier` must have relationship-*Exclusive* for access *exclusive* operations.

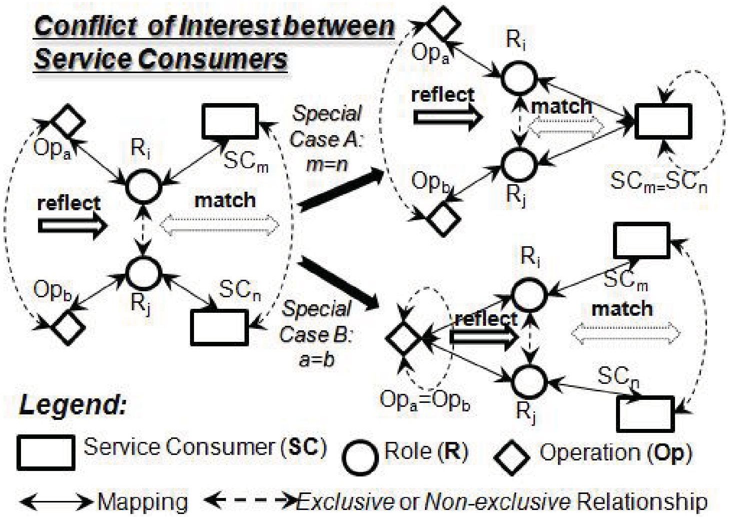

Fig. 3. Role Relationship Check (R-RC)

Let us take another example. "Double Check" policy is enforced in Tom & Brothers on operation of *Payment Verification*, i.e., two financial institutes are required to ensure the payment from purchaser. In order to avoid fraudulent payment assessment on purchase, an *exclusive* relationship between the `Initial Verifier` and `Second Verifier` must be enforced. Westpac Financial Service and St. George Financial Consultant Company are two financial institutes with *non-exclusive* relationship because they belong to the same financial group. Westpac Financial Service and St. George Financial Consultant Company can not be assigned the roles `Initial Verifier` and `Second Verifier` to do payment verification for one transaction due to their *non-exclusive* relationship.

To prevent conflict of interest among service consumers, the following authorization rule named as Static Role Relationship Check (**S-R-RC**) is specified as follows:

Authorization Rule 1. S-R-RC: *Let* $\mathbb{SC}$ *be a set of Service Consumers. Let* $\mathbb{R}$ *be a set of Roles. We say that, there is no conflict of interest between service consumers, formally* $\exists r_i \in \mathbb{R}, \exists sc_m \in \mathbb{SC}, (r_i, sc_m) \in SCA$*, if there exists a subset of Role* $\mathbb{R}$ *named as* $\widetilde{R_a}$*, which includes all roles that have been mapped with service consumers, and the relationships between* r_i *and roles in* $\widetilde{R_a}$ *are the same as the relationships between* sc_m *and service consumers that have been mapped to the roles in* $\widetilde{R_a}$*. Formally,* $\exists \widetilde{R_a} \subseteq \mathbb{R}\text{-}\{r_i\}$, $\forall\, r_j \in \widetilde{R_a}$, $(r_j,\, assigned\_sc(r_j)) \in SCA$, $\forall r'_j \in \mathbb{R}\text{-}\{r_i\}\text{-}\widetilde{R_a}$, $assigned\_sc(r'_j) = \emptyset$, $\mathcal{RL}(r_i, r_j) = \mathcal{RL}(sc_m, assigned\_sc(r_j))$*, where* $\mathcal{RL}(element,\ element)=\{\otimes,\ \ominus\}$ *reflecting the exclusive, or non-exclusive relationships between elements.*

As an alternative solution, roles can be assigned without using the above authorization rule but the conflict of authorization between consumers will be checked at run time. The mapping between service consumers and roles can be stored in the system at design time. The conflict of interest between consumers are checked when the assigned roles are activated simultaneously by a specific consumer. The authorization rule named as Dynamic Role Relationship Check (**D-R-RC**) is specified as follows:

Authorization Rule 2. D-R-RC: *Let $\mathbb{SC}$ be a set of Service Consumers. Let $\mathbb{R}$ be a set of Roles. We say that, there is no runtime conflict of interest between service consumers, formally $\exists r_i \in \mathbb{R}, \exists sc_m \in \mathbb{SC}, (r_i, sc_m) \in SCA$, and $r_i \in RSi(SCSi(sc_m))$ and/or $r_i \in RSc(SCSc(sc_m))$, if there exists a subset of Role $\mathbb{R}$ named as $\widetilde{R_b}$, which includes all roles that are being activated; the relationships between r_i and all roles in $\widetilde{R_b}$ are the same as the relationships between sc_m and service consumers that are activating these roles in $\widetilde{R_b}$. Formally, $\exists\widetilde{R_b} \subseteq \mathbb{R}$-$\{r_i\}$, $\forall r_k \in \widetilde{R_b}$, $\exists sc_n \in \mathbb{SC}$, $r_k \in RSi(SCSi(sc_n))$ and/or $r_k \in RSc(SCSc(sc_n))$, $\forall r'_k \in \mathbb{R}$-$\widetilde{R_b}$-$r_i$, $\forall sc'_n \in \mathbb{SC}$, $r'_k \notin RSi(SCSi(sc'_n))$, and $r'_k \notin RSc(SCSc(sc'_n))$, $\mathcal{RL}(r_i, r_k) = \mathcal{RL}(sc_m, sc_n)$.*

3.2 Conflict of Interest between Resources

If two resources have the relationship *Exclusive* or *Non-exclusive*, the mapped resource types for these two resources must have the same relationship as *exclusive* or *non-exclusive* to prevent conflict of interest.

In Fig.4, if Op_a and Op_b have *exclusive* relationship ($Op_a \otimes Op_b$), then the relevant resource type ReT_i and ReT_j should be *exclusive* ($Op_a \otimes Op_b \Rightarrow ReT_i \otimes RetT_j$). The relationship between the resource types mapped with resources Re_k and Re_h must be the same as the relationship between these two resources. If the resources Re_k and Re_h are *non-exclusive* with each other ($Re_k \ominus Re_h$), e.g., belonging to one company group, then Re_k and Re_h can not be mapped to resource type ReT_i and ReT_j respectively at the same time, since ReT_i and ReT_j are *exclusive*. To avoid conflict of interest, two resources with relationship $\otimes$ or $\ominus$ must be included in the associated two resource types with the same relationship $\otimes$ or $\ominus$.

Two special cases are described in Fig.4, where operations (Special Case A in Fig.4) and resources (Special Case B in Fig.4) become one operation and one resource respectively. Let us take an example as special case B in Fig. 4. For the security reason, *Order Engine* and *Order Engine Accessory* are *exclusive* operations in Tom & Brothers (particularly for military customer), where the mapped resource type, `Engine Supplier` and `Engine Accessory Supplier`, are *exclusive*. Hence, if the resource mapped to `Engine Supplier` and `Engine Accessory Supplier` are the same one, *non-exclusive* relationship exists between the resource and itself, and the resource can not be included in resource type `Engine Supplier` and `Engine Accessory Supplier` at the same time.

We devise the authorization rule named as Static Resource Type Relationship Check (**S-RT-RC**) on the mapping of resources and resource types to prevent the conflict of interest between resources. Here we formally define the authorization rule at design time as follows:

Authorization Rule 3. S-RT-RC: *Let $\mathbb{R}e$ be a set of Resources. Let $\mathbb{R}e\mathbb{T}$ be a set of Resource Types. We say that, there is no conflict of interest between*

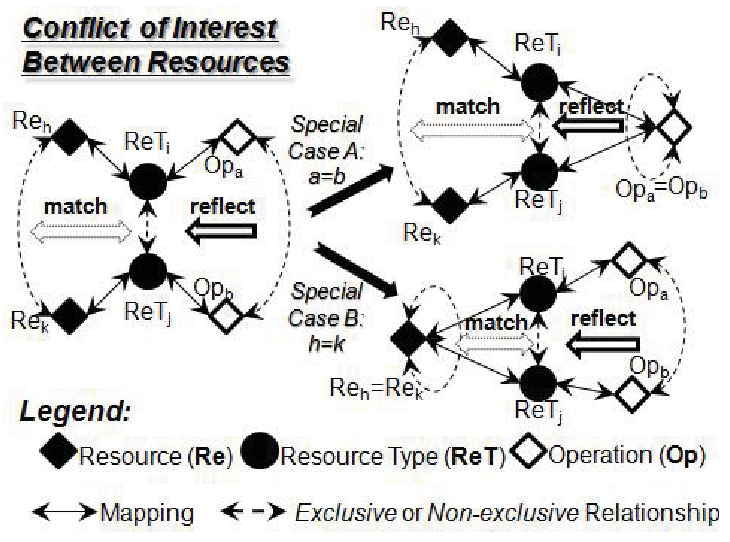

Fig. 4. Resource Type Relationship Check (RT-RC)

resources, formally $\exists ret_i \in \mathbb{R}e\mathbb{T}, \exists re_h \in \mathbb{R}e, (re_h, ret_i) \in RTA$, *if there exists a subset of* $\mathbb{R}e\mathbb{T}$ *named as* $\widetilde{ReT_a}$ *that includes all resource types that have been mapped with resources, and the relationships between* ret_i *and resource types in* $\widetilde{ReT_a}$ *are the same as the relationships between* re_h *and the resources mapped with resource types in* $\widetilde{ReT_a}$. *Formally* $\exists \widetilde{ReT_a} \subseteq \mathbb{R}e\mathbb{T}\text{-}\{ret_i\}$, $\forall\ ret_j \in \widetilde{ReT_a}$, $(ret_j,\ assigned\_re(ret_j)) \in RTA$, $\forall ret'_j \in \mathbb{R}e\mathbb{T}\text{-}\{ret_i\}\text{-}\widetilde{ReT_a}$, $assigned\_re(ret'_j) = \emptyset$, $\mathcal{RL}(ret_i, ret_j) = \mathcal{RL}(re_h, assigned\_re(ret_j))$.

Alternatively, the resource can be mapped to resource types without using the above authorization rule, but the conflict of interest between resources will be checked at run time. The mapping between resources and resource types can be stored in system at design time. The conflict of interest between resources are checked when the resource types are activated simultaneously by employing the resources to provide support to the operations. Here we formally define the Dynamic Resource Type Relationship Check (**D-RT-RC**) on preventing runtime conflict of interest between resources.

Authorization Rule 4. D-RT-RC: *Let* $\mathbb{R}e$ *be a set of Resources. Let* $\mathbb{R}e\mathbb{T}$ *be a set of Resource Types. We say that, there is no runtime conflict of interest between resources, formally* $\exists ret_i \in \mathbb{R}e\mathbb{T}, \exists re_h \in \mathbb{R}e, (re_h, ret_i) \in RTA$, *and* $ret_i \in RTS(RES(re_h))$, *if there exists a subset of* $\mathbb{R}e\mathbb{T}$ *named as* $\widetilde{ReT_b}$ *includes all resource types that are being activated; The relationships between* ret_i *and resource types in* $\widetilde{ReT_b}$ *should be the same as the relationships between* re_h *and resources that are employed to support operations by specific resource types in* $\widetilde{ReT_b}$. *Formally,* $\exists \widetilde{ReT_b} \subseteq \mathbb{R}e\mathbb{T}\text{-}\{ret_i\}$, $\forall ret_k \in \widetilde{ReT_b}$, $\exists re_l \in \mathbb{R}e$, $ret_k \in RTS(RES(re_l))$, $\forall ret'_k \in \mathbb{R}e\mathbb{T}\text{-}\widetilde{ReT_b}\text{-}\{ret_i\}$, $\forall re'_l \in \mathbb{R}e$, $ret'_k \notin RTS(RES(re'_l))$, $\mathcal{RL}(ret_i, ret_k) = \mathcal{RL}(re_h, re_l)$.

3.3 Conflict of Interest between Service Consumers and Resources

Resources and service consumers can have relationship as *exclusive* or *non-exclusive* that must be the same as the relationship of mapped roles and resource types. The relationship between a resource type and a role reflects the relationship between the operation that the role need to access and the operation that the resource type can support. The conflict of interest between service consumers and resources can occur, if the relationship between the service consumers and the resources is not the same as the relationship of mapped role and resource type.

Two special case are also presented in Fig. 5, where (1) the operation that the role need to access and the operation that the resource type can support are the same one (Special Case A in Fig. 5), and (2) the service consumer and the resource are the same web service (Special Case B in Fig. 5). In special case A at Fig. 5, the operation that the resource type ReT_j supports is what the role R_j need to access ($Op_a = Op_b$). Their relationship is *non-exclusive* ($Op_a \ominus Op_b$). If the relationship between the mapped service consumer SC_m and resource Re_k is *exclusive* ($SC_m \otimes Re_k$), e.g., the Chinese manufactory as the resource and the USA military customer as the service consumer, the mapping between the service consumer SC_m to the specific role R_j and the mapping between the resource Re_k to the specific resource type ReT_j can not be made simultaneously.

Let us take another example, in special case B at Fig. 5, a service consumer and a resource belong to one web service ($SC_m = Re_k$). Their relationship is *non-exclusive* ($SC_m \ominus Re_k$). If the operation that the web service supports as resource is *exclusive* with the operation that the web service need to access as the service consumer ($Op_a \otimes Op_b$), there is a conflict of interest between the consumer and the resource. If the web service is assigned with specific role R_i to access the operation Op_a, it can not be mapped to resource type ReT_j to support operation Op_b; vise versa.

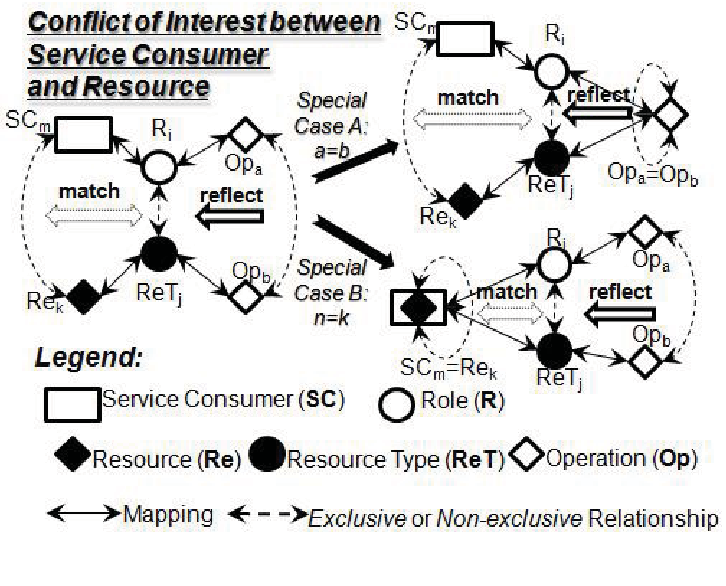

Fig. 5. Role & Resource Type Relationship Check (RRT-RC)

We define authorization rule named as Static Role & Resource Type Relationship Check (S-RRT-RC) to prevent the conflict of interest between the consumer and the resource. The formal specification is as follows:

Authorization Rule 5. S-RRT-RC: *Let $\mathbb{R}e$ be a set of Resources, and $\mathbb{SC}$ be a set of Service Consumers. Let $\mathbb{R}$ be a set of Roles, and $\mathbb{R}e\mathbb{T}$ be a set of Resource Types. We say that, there is no conflict of interest between service consumer and resource if (1) and (2) are satisfied:*

1. *service consumer sc_m and role r_i can be mapped in SCA, formally $\exists r_i \in \mathbb{R}, \exists sc_m \in \mathbb{SC}, (r_i, sc_m) \in SCA$, if there exists a set named as $\widetilde{ReT_a}$ that is a subset of Resource Types and includes all resource types that have been mapped with specific resources; The relationships between r_i and resource types in $\widetilde{ReT_a}$ should be the same as the relationships between sc_m and the resources that are mapped with the resource types in $\widetilde{ReT_a}$. Formally, $\exists \widetilde{ReT_a} \subseteq \mathbb{R}e\mathbb{T}$, $\forall\ ret_j \in \widetilde{ReT_a}$, $(ret_j,\ assigned\_re(ret_j)) \in RTA$, $\forall ret'_j \in \mathbb{R}e\mathbb{T} - \widetilde{ReT_a}$, $assigned\_re(ret'_j) = \emptyset$, $\mathcal{RL}(ret_j, r_i) = \mathcal{RL}(assigned\_re(ret_j), sc_m)$.*
2. *resource type ret_i and resource re_h can be mapped in RTA, formally $\exists ret_i \in \mathbb{R}e\mathbb{T}, \exists re_h \in \mathbb{R}e, (re_h, ret_i) \in RTA$, if there exists a set R_a as a subset of Roles that includes all roles which have been assigned to specific service consumers, and the relationships between ret_i and all roles in R_a should be the same as the relationship between re_h and service consumers that are assigned as specific roles in R_a. Formally, $\exists \widetilde{R_a} \subseteq \mathbb{R}$, $\forall\ r_j \in \widetilde{R_a}$, $(r_j,\ assigned\_sc(r_j)) \in SCA$, $\forall r'_j \in \mathbb{R} - \widetilde{R_a}$, $assigned\_sc(r'_j) = \emptyset$, $\mathcal{RL}(ret_i, r_j) = \mathcal{RL}(re_h, assigned\_sc(r_j))$.*

The mappings between roles and service consumers, and the mappings between resources and resource types can be made without using the above authorization rule. The conflict of interest between service consumers and resources will be checked at runtime. The mappings between role and service consumer, and the mapping between resource and resource type can be stored in system at design time. The conflict of interest between service consumer and resource is checked when the assigned role and resource type are activated simultaneously in the execution of the composite web service requested by the specific service consumer. The authorization rule named as Dynamic Role & Resource Type Relationship Check (D-RRT-RC) is specifies as follows:

Authorization Rule 6. D-RRT-RC *Let $\mathbb{R}e$ be a set of Resources, and $\mathbb{SC}$ be a set of Service Consumers. Let $\mathbb{R}$ be a set of Roles, and $\mathbb{R}e\mathbb{T}$ be a set of Resource Types. We say that, there is no runtime conflict of interest between service consumer and resource if (1) and (2) are satisfied:*

1. *service consumer sc_m can activate assigned role r_i, formally $\exists r_i \in \mathbb{R}, \exists sc_m \in \mathbb{SC}, (r_i, sc_m) \in SCA$, and $r_i \in RSc(SCSc(sc_m))$, if there exists a subset of Resource Types named as $\widetilde{ReT_b}$ that includes all resource types that are activated (when resources mapped to these resource types are required to provide support to operations); the relationship between r_i and all resource types*

in $\widetilde{ReT_b}$ should be the same as the relationship between sc_m and resources that are activating the resource types in $\widetilde{ReT_b}$. Formally, $\exists \widetilde{ReT_b} \subseteq \mathbb{R}e\mathbb{T}$, $\forall ret_g \in \widetilde{ReT_b}$, $\exists re_h \in \mathbb{R}e$, $ret_g \in RTS(RES(re_h))$, $\forall ret'_g \in \mathbb{R}e\mathbb{T} - \widetilde{ReT_b}$, $\forall re'_h \in \mathbb{R}e$, $ret'_g \notin RTS(RES(re'_h))$, $\mathcal{RL}(ret_g, r_i) = \mathcal{RL}(re_h, sc_m)$.

2. *resource type ret_i is being activated by resource re_h, formally $\exists ret_i \in \mathbb{R}e\mathbb{T}$, $\exists\ re_h \in \mathbb{R}e, (re_h, ret_i) \in RTA$, and $ret_i \in RTS(RES(re_h))$, if there exists a subset of $\mathbb{R}$ named as $\widetilde{R_b}$ that includes all of roles that have been activated; The relationship between ret_i and roles in $\widetilde{R_b}$ should be the same as the relationship between re_h and service consumers that activate the roles in $\widetilde{R_b}$. Formally, $\exists \widetilde{R_b} \subseteq \mathbb{R}$, $\forall r_g \in \widetilde{R_b}$, $\exists sc_m \in \mathbb{SC}$, $r_g \in RSc(SCSc(sc_m))$, $\forall r'_g \in \mathbb{R}$-$\widetilde{R_b}$, $\forall sc'_m \in \mathbb{SC}$, $r'_g \notin RSc(SCSc(sc'_m))$, $\mathcal{RL}(ret_i, r_g) = \mathcal{RL}(re_h, sc_m)$.*

Conflict of interest between one pair of service consumer/resource and other pairs of service consumer/resource is another new type of conflict of interest which can be identified in SOAC. A service consumer and a resource is put in one pair when the service consumer request the access of the operation of a composite web service and the operation needs the support of the resource. The relationships between pairs of role/resource type reflect the relationships between operations mapped to these pairs of role/resource type. If two pairs of service consumer/resource have the relationship *Exclusive* or *Non-exclusive*, the pairs of mapped roles and resource types must have the same relationship as *Exclusive* or *Non-exclusive*

For example, in Fig. 6, if the operations are *exclusive* ($Op_a \otimes Op_b$), the relationship between the pairs of mapped roles and resource types must also be *exclusive* ($Op_a \otimes Op_b \Rightarrow (R_i, ReT_i) \otimes (R_j, ReT_j)$). Note, here the relationship between operations will be reflected by the relationship between the pairs of roles and resource types rather than considering the relationship between roles or resources types individually which are discussed in previous subsections 3.1 and 3.2. If the relationship between two pairs of service consumer and resource are *non-exclusive* ($(SC_n Re_h) \ominus (SC_m, Re_k)$), the pairs of mapped roles and resource types must also be *non-exclusive* to prevent the conflict of interest.

Two special cases are illustrated in Fig. 6, where (1) the pairs of service consumer and resource are the same one (Special case A in Fig. 6 ($SC_m = SC_n$ and $Re_h = Re_k$)), and (2) the operations in different authorizations are the same one (Special cased B in Fig. 6 ($Op_a = Op_b$)). Let us take an example in special case A. When a service consumer is mapped with the role `Military Customer` by Tom & Brothers, and the goods it orders need to be supplied by part manufactory mapped with resource type `Vehicle Engine Supplier`, it will violate the law if Tom & Brothers also use the same manufactory that is mapped with resource type `Vehicle Engine Accessory Supplier` to supply the engine accessory to the same consumer that is mapped with role `Commercial Customer`. In this case, the *exclusive* relationship between operations of *Order Engine* and *Order Engine Accessory* requires that the relationship between the pair of `Military Customer` and `Vehicle Engine Supplier` and the pair of

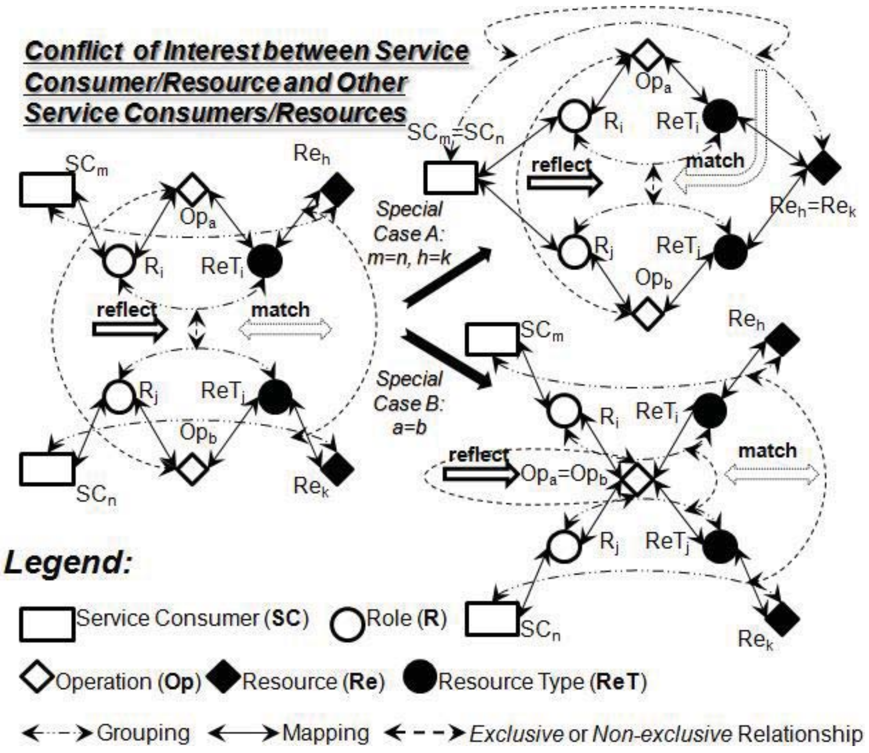

Fig. 6. Pairs of Role & Resource Type Relationship Check (PRRT-RC)

`Commercial Customer` and `Vehicle Engine Accessory Supplier` are *exclusive* also. If the two pairs of role and resource type are mapped to the same pair of service consumer and resource with a *non-exclusive* relationship, the conflict of interest occurs. This conflict of interest is identified to prevent the following two things happening at the same time. The first thing is to assemble the engine for military use with the engine accessory for civil use and the second thing is to purchase engine and engine accessory from the same part suppliers. We can observe that the service consumer can be mapped with both roles `Military Customer` and `Commercial Customer` without causing conflict of interest between customers (discussed in subsection 3.1). We can also observe that the manufactory can be mapped with both resource types `Vehicle Engine Supplier` and `Vehicle Engine Accessory Supplier` without causing conflict of interest between resources (discussed in subsection 3.2). The conflict of interest occurs when the service consumer is mapped with both roles and the resource is mapped with both resource types. In a summary, if the manufactory as `Vehicle Engine Supplier` to provide engine to a service consumer as `Military Customer`, it should not provide engine accessory to the same service consumer that is identified as `Commercial Customer`; vise versa.

We set up authorization rule named as Static Pairs of Role & Resource Type Relationship Check (S-PRRT-RC) to prevent conflict of interest between two pairs of service consumer/resource. Here we formally define the authorization rule at design time as follows:

Authorization Rule 7. S-PRRT-RC *Let $\mathbb{R}e$ be a set of Resources, and $\mathbb{SC}$ be a set of Service Consumers. Let $\mathbb{R}$ be a set of Roles, and $\mathbb{R}e\mathbb{T}$ be a set of Resource Types. We say that, there is no conflict of interest between one pair of service consumer/resource and another pair of service consumer/resource, formally $\exists r_i \in \mathbb{R}, \exists sc_m \in \mathbb{SC}, (r_i, sc_m) \in SCA$, $\exists ret_i \in \mathbb{R}e\mathbb{T}, \exists re_h \in \mathbb{R}e$, $(re_h, ret_i) \in RTA$, $assigned\_op(r_i) \cap$ $assigned\_ret(ret_i) \neq \emptyset$, if there exists a set named as $\widetilde{ReT_a}$ that is a subset of Resource Types and includes all resource types which have been mapped with specific resources, and exists a set named as $\widetilde{R_a}$ that is a subset of Roles and includes all roles which have been assigned to specific service consumers; There must exist a resource type (ret_k) and a role (r_k) that map to the same operations; the relationship between (ret_i, r_i) and (ret_k, r_k) should be the same as the relationships between (re_h, sc_m) and pairs of resources and service consumers that are mapped to ret_k and r_k respectively. Formally, $\exists \widetilde{ReT_a} \subseteq \mathbb{R}e\mathbb{T}$, $\forall\ ret_j \in \widetilde{ReT_a}$, $(ret_j,\ assigned\_re(ret_j)) \in RTA$, $\forall ret'_j \in \mathbb{R}e\mathbb{T} - \widetilde{ReT_a}$, $assigned\_re(ret'_j) = \emptyset$, $\exists \widetilde{R_a} \subseteq \mathbb{R}$, $\forall\ r_j \in \widetilde{R_a}$, $(r_j,$ $assigned\_sc(r_j)) \in SCA$, $\forall r'_j \in \mathbb{R} - \widetilde{R_a}$, $assigned\_sc(r'_j) = \emptyset$, $\exists r_k \in \widetilde{R_a}, \exists ret_k \in \widetilde{ReT_a}$, $assigned\_op(r_k) \cap assigned\_ret(ret_k) \neq \emptyset$, $\mathcal{RL}((ret_i, r_i), (ret_k, r_k)) = \mathcal{RL}$ $((re_h, sc_m), (assigned\_re(ret_k), assigned\_sc(r_k)))$.*

Without using the above authorization rule, the conflict of interest between pairs of service consumer and resource can be checked at runtime. The mapping between service consumer and role, and the mapping between resource type and resource are stored in system. The conflict of interest between pairs of service consumer/resource is checked when the associated roles and resource types are activated simultaneously. The authorization rule named as Dynamic Pairs of Role & Resource Type Relationship Check (D-PRRT-RC) is specified as follows:

Authorization Rule 8. D-PRRT-RC *Let $\mathbb{R}e$ be a set of Resources, and $\mathbb{SC}$ be a set of Service Consumers. Let $\mathbb{R}$ be a set of Roles, and $\mathbb{R}e\mathbb{T}$ be a set of Resource Types. We say that, there is no runtime conflict of interest between one pair of service consumer/resource and another pair of service consumer/resource, formally $\exists r_i \in \mathbb{R}, \exists sc_m \in \mathbb{SC}, (r_i, sc_m) \in SCA$, $\exists ret_i \in \mathbb{R}e\mathbb{T}, \exists re_h \in \mathbb{R}e, (re_h, ret_i) \in RTA$, $assigned\_op(r_i) \cap assigned\_ret(ret_i) \neq \emptyset$, $r_i \in RSc\ (SCSc(sc_m))$, and $ret_i \in RTS(RES(re_h))$, if there exists a subset of $\mathbb{R}e\mathbb{T}$ named as $\widetilde{ReT_b}$ which includes all resource types that are being activated, and there also exists a subset of $\mathbb{R}$ named as $\widetilde{R_b}$ which includes all roles that are being activated. There must exists a resource type ret_k belonging to $\widetilde{ReT_b}$ and a role r_k belonging to $\widetilde{R_b}$ that are supporting and accessing the same operations respectively; The relationship between (ret_i, r_i) and (ret_k, r_k) should be the same as the relationship between (re_h, sc_m) and pairs of resources and service consumers that are activating ret_k and r_k respectively. Formally $\exists \widetilde{ReT_b} \subseteq \mathbb{R}e\mathbb{T}$, $\forall ret_x \in \widetilde{ReT_b}$, $\exists re_y \in \mathbb{R}e$, $ret_x \in RTS(RES(re_y))$, $\forall ret'_x \in \mathbb{R}e\mathbb{T} - \widetilde{ReT_b}$, $\forall re'_y \in \mathbb{R}e$, $ret'_x \notin RTS(RES(re'_y))$, $\exists \widetilde{R_b} \subseteq \mathbb{R}$, $\forall r_x \in \widetilde{R_b}$, $\exists sc_y \in \mathbb{SC}$, $r_x \in RSc(SCSc(sc_y))$, $\forall r'_x \in \widetilde{R_b}$, $\forall sc'_y \in \mathbb{SC}$, $r'_x \notin RSc(SCSc(sc'_y))$, $\exists r_k \in \widetilde{R_b}, \exists ret_k \in \widetilde{ReT_b}$, $assigned\_op(r_k) \cap assigned\_ret(ret_k)$*

$\neq \emptyset$, $\exists sc_n \in \mathbb{SC}$, $r_k \in RSc(SCSc(sc_n))$, $\exists re_g \in \mathbb{R}e$, $ret_k \in RTS(RES(re_g))$, $\mathcal{RL}((ret_i, r_i), (ret_k, r_k)) = \mathcal{RL}((re_h, sc_m), (re_g, sc_n))$.

4 Related Work

Role based access control [2, 3] is a widely accepted approach to restrict system access to authorized users. In RBAC, users acquire permissions through their roles rather than they are assigned permissions directly. Traditional RBAC models deal with authorization of resources which belong to an individual organization. In web service paradigm, the component services of composite web services and their related resources normally spread over multiple organizations and are invoked in a highly dynamic manner. Traditional RBAC models can not be used directly as ready solutions for authorization of web services. There have been quite a lot of researches about authorization of web services. We will overview some representative work in the follows.

In [5, 6], the authors propose a RBAC framework to manage access control in WS-BPEL [7], named RBAC-WS-BPEL. In RBAC-WS-BPEL, authorization constrains are specified on the execution activities and roles are assigned to users for gaining permissions on execution activities. This research only focuses on the service orchestration level and it has no capability to consider characteristics of resources required by composite web services.

In [8, 9], the authors provide an enforcement and verification approach to guarantee that a service choreography can be successfully implemented between a set of web services (service consumer and the composite web service) based on their authorization constraints. This research only focuses on enforcement and verification of authorization between service consumer and composite web service. The component web services in authorization of composite web service are not taken into account.

An access control model CWS-RBAC was proposed in [10] which takes the composite service into consideration and is comparable to our proposed approach. In CWS-RBAC, a global role is assigned to service consumers to gain the permission to access the composite service and a local role mapped from global role is assigned to service consumer to access the other component services. The authors in [11] propose another concept-*Role Composition* where global role and local role are composed together. It analyzes how a local role issued by an individual component service is mapped to a global role from the composite services. In that case, if the service consumer is assigned with a global role, then it automatically bears the permissions of the bound local role on the component service. In these approaches, the "role" as a concept used by a specific service to manage the authorization is part of internal security policy within an individual web service and can not be identified by other services. For example, the composite web service can not identify which role that it can be assigned by the component web service that it needs to access. Actually, the composite web service can only perceive the permissions based on credentials, i.e, the authorization constraints (the public part of authorization policy of each web service). Hence, the mapping of the global role issued from the composite service with the local role generated in other component services is not realistic. In our proposed approach, we introduce the resource type (**ReT**) to explicitly express characteristics and requirements of resources associated with component services. **ReT** can support an efficient way for the management of dynamically involved and prone-to-change component services in composite web services. Furthermore, **ReT** provides the fundamental concept for defining the conflict of interest related with component services.

Conflict of interest is a major concern in traditional RBAC models. In order to deal with conflict of interest, static and dynamic separation of duty mechanisms are defined in RBAC standard [12, 14]. The authors in [13] have discussed the conflict of interest in the authorization of web services. However, this research deals with the authorization of web services using the same way as those authorizations in close systems. In particular, the features of composite web services have not been taken into consideration. It is lacking of existing work to identify and deal with possible types of conflict of interest among service consumers, among component services, and between service consumers and component services in composite web services.

Existing approaches about authorization of composite web services have the limitations: (1) ignoring the dynamic nature of composite web services that require resources based on-demand; (2) missing an efficient way to the administration of the resources in service-oriented authorization; (3) hard coding the roles issued from resources and composite service; and (4) lacking of authorization rules for preventing conflict of interest in composite web service authorization. This paper reports our research for the authorization of composite web services to address the above mentioned limitations of existing approaches.

5 Conclusion and Future Work

The proposed approach for authorization of composite web services can provide an efficient way to administrate and manage a large number of service consumers and dynamic component services. This research addresses the conflict of interest issue regarding to both service consumers and component services in composite web services. Four types of conflict of interest are identified. Authorization rules at both design time and run time to deal with various types of conflict of interest are provided and illustrated. In the future, we plan to investigate the possibility of employing the hierarchical structure to represent resource types and the mechanism of mapping between resource types and individual resources.

References

[1] Papazoglou, M., Georgakopoulos, D.: Service-Oriented Computing. Communications of the ACM 46(10), 25–28 (2003)

[2] Sandhu, R.S., Coyne, E., Feinstein, H., Youman, C.: Role-based Access Control Models. IEEE Computer 29(2), 38–47 (1996)

[3] Ferraiolo, D., Cugini, J., Kuhn, R.: Role Based Access Control: Features and Motivations. In: Proceedings of ACSAC (1995)

[4] Sun, H., Zhao, W., Yang, J.: SOAC: A Conceptual Model for Managing Service-Oriented Authorization. In: Proceedings of the IEEE International Conference on Service Computing, pp. 546–553 (2010)

[5] Bertino, E., Crampton, J., Paci, F.: Access Control and Authorization Constraints for WS-BPEL. In: Proceedings of the IEEE International Conference on Web Services, pp. 275–284 (2006)

[6] Paci, F., Bertino, E., Crampton, J.: An Access Control Framework for WS-BPEL. International Journal of Web Service Research 5(3), 20–43 (2008)

[7] Jordan, D., et al.: Web Services Business Process Execution Language Version 2.0 (WS-BPEL 2.0) (August. 2006), http://docs.oasis-open.org/wsbpel/2.0/

[8] Mecella, M., Ouzzani, M., Paci, F., Bertino, E.: Access Control Enforcement for Conversation-based Web Service. In: Proceedings of the International World Wide Web Conference, pp. 257–266 (2006)

[9] Paci, F., Ouzzani, M., Mecella, M.: Verification of Access Control Requirements In Web Servies Choreography. In: Proceedings of SCC, pp. 5–12 (2008)

[10] Wonohoesodo, R., Tari, Z.: A Role Based Access Control for Web Services. In: Proceedings of SCC, pp. 49–56 (2004)

[11] Fischer, J., Majumdar, R.: A Theorey of Role Composition. In: Proceedings of ICWS, pp. 49–56 (2008)

[12] Ferraiolo, D., Sandhu, R., et al.: Proposed NIST Standard for Role-Based Access Control. ACM Trans. on Information and System Security (TISSEC) 4(3), 224–274 (2001)

[13] Giblin, C., Hada, S.: Towards Separation of Duties for Services. In: The 6th Int. Workshop on SOA & Web Services Best Practices Committee, OOPSLA, Nashville, October 19 (2008)

[14] Ahn, G., Sandhu, R.: Role-Based Authorization Constraints Specification. ACM Transactions on Information and System Security (TISSEC) 3(4), 207–226 (2000)