23

A SERVICE COMPOSABILITY MODEL TO SUPPORT DYNAMIC COOPERATION OF CROSS-ENTERPRISE SERVICES

Jing-fan Tang
College of Computer Science & Technology, Hangzhou Dianzi University,
tangjf@hziee.edu.cn

With the development of web services related technologies, more and more enterprises adopt web services to encapsulate their business systems to be published on Internet. Due to the different semantic of the web services, it brings much difficulty to implement the dynamic cooperation of the cross-enterprise services efficiently. This paper introduces a service composability model to support the dynamic cooperation of cross-enterprise services by utilizing the semantic rules of the internal composability. The WSDL descriptions of the web services are extended with semantic capability, which enable the cross-enterprise services to be composed and cooperated automatically according to the synthetic comparison of the semantic features. Furthermore, a policy driven negotiation process is also proposed to enable the cooperation of cross-enterprise services to achieve win-win.

1. INTRODUCTION

With the rapid development of technologies on web services and service-oriented architecture, more and more enterprises utilize web services to encapsulate the internal business process to provide web applications for all kinds of clients. The cooperation process of cross-enterprise services can be seen as services composition process. A web service will find a composable service to be composed and achieve cooperation. The big challenge for composing the cross-enterprise services is to enable the description of each service to be understood by each other, such as the functionalities, interfaces, etc.

The semantics of web services is crucial to enabling automatic service composition. It is important to insure that selected services for composition offer the "right" features. Such features may be syntactic (e.g., number of parameters included in a message sent or received by a service). They may also be semantic (e.g., the business functionality offered by a service operation or the domain of interest of the service). To help capture Web services' semantic features, the concept of ontology is used. Ontology is a shared conceptualization based on the semantic proximity of terms in a specific domain of interest (McIlraith, 2001). Ontologies are increasingly seen as key to enabling semantics-driven data access and processing (Bussler, 2002). They are expected to play a central role in the Semantic Web, extending syntactic service interoperability to semantic interoperability (Horrocks, 2002).

In this paper, we introduce a service composability model to support the dynamic cooperation of cross-enterprise service by utilizing the semantic rules of the internal composability. Through the composability rules registered in an enhanced UDDI center, an automatic matchmaking process will be executed to find out the candidate services for the cooperation. In order to achieve the win-win cooperation among the cross-enterprise services, a policy driven negotiation process is adopted to choose the optimum service to establish the final cooperation.

The rest of the paper is organized as following. Section 2 introduces the related work. Section 3 describes the service composability model. Section 4 discusses the automatic matchmaking of the cross-enterprise services through the composability rules. Section 5 introduces the negotiation process, which is policy-driven. Section 6 concludes.

2. RELATED WORK

There has been some research on the service composition to support cross-enterprise business integration and cooperation (Vaggelis, 1999; Yang, 2001; Alan, 2002). And, the automatic composition of web services is a recent trend (McIlraith, 2001; Berners-Lee, 2001). This would include the automatic selection and interoperation of web services. Automatic composition is slated to play a major role in enabling the envisioned Semantic Web (Weikum, 2002).

A novel approach has been presented in (Zhou, 2005)to support the dynamic establishment of virtual enterprise based on the dynamic web service composition. A service execution plan will be dynamically built to aim at the target of the virtual enterprise through the composition of web services. The development and deployment of web services is separated to support the dynamically deploying and binding of the web services at run time. The members in the virtual enterprise will be located through the dynamic web service discovery based on DAML+OIL logic reasoning, and selected through the dynamic web services negotiation with multi-steps protocol to organize the virtual enterprise at run time.

Based on the research in (Zhou, 2005; Brahim, 2003; Tang, 2004a; Tang, 2004b; Tang, 2005], we will introduce a service composability model to support dynamic cooperation of cross-enterprise services in this paper.

3. SERVICE COMPOSIBILITY MODEL

3.1 Semantic description of web services

In order to support dynamic cooperation of cross-enterprise services, the description of each service should be understood by other services. An emerging language for describing operational features of Web services is WSDL (Web Service Description Language). WSDL is being standardized within the W3C consortium. WSDL has gained considerable momentum as the language for web service description. However, WSDL provides little or no support for semantic description of Web services. It mainly includes constructs that describe web services from a syntactic point of view. To cater to semantic web-enabled web services, we should extend WSDL with semantic capabilities, which would lay the groundwork for the automatic selection and cooperation of cross-enterprise web services.

Before introducing the semantic description of web services, we give some definitions as following (Brahim, 2003):

Definition 1 – *Message*. A *message M* is defined as a tuple (P, T, U, R) where:

- *P* is a set of parameter names.
- *T*: *P* $\rightarrow$ *DataTypes* is a function that assigns a data type to each parameter. *DataTypes* is a set of XML data types.
- *U*: *P* $\rightarrow$ *Units* is a function that gives the unit of measurement used for each parameter. *Units* is a taxonomy for measurement units.
- *R*: *P* $\rightarrow$ *Roles* is a function that assigns a business role to each parameter. *Roles* is a taxonomy for business roles.

Definition 2 – *Purpose*. The *purpose* of an operation op_{ik} is defined by a tuple *(Function, Synonyms, Specialization)* where *Function* is op_{ik}'s business functionality defined within a given taxonomy, *Synonyms* is a set of alternative function names, and *Specialization* is a set of characteristics of op_{ik}'s function.

Definition 3 – *Category*. The *category* of an operation op_{ik} is defined by a tuple *(Domain, Synonyms, Specialization)* where *Domain* is op_{ik}'s area of interest defined within a given taxonomy, *Synonyms* is a set of alternative domains, and *Specialization* is a set of characteristics of op_{ik}'s domain.

Definition 4 – *Quality*. The *quality* of an operation op_{ik} is defined by a tuple $(Fees_{ik}, Security_{ik}, Privacy_{ik})$. $Fees_{ik}$ is the dollar amount needed to execute op_{ik}. $Security_{ik}$ is a boolean that specifies whether op_{ik}'s messages are securely exchanged. $Privacy_{ik}$ is the set of input and output parameters that are not divulged to external entities.

Definition 5 – *Operation*. An operation op_{ik} is defined by a tuple $(Description_{ik}, Mode_{ik}, In_{ik}, Out_{ik}, Purpose_{ik}, Category_{ik}, Quality_{ik})$ where:

- $Description_{ik}$ is a text summary about the operation features.
- $Mode_{ik} \in$ {"one-way", "notification", "solicit-response", "request-response"}.
- In_{ik} and Out_{ik} are the input and output messages, respectively. $In_{ik} = (\Phi, \Gamma_{ik})$ and $Out_{ik} = (\Phi, \Gamma_{ik})$ for notification and one-way operations, respectively.
- $Purpose_{ik}$ describes the business function offered by the operation (cf. Definition 2).
- $Category_{ik}$ describes the operation's domain of interest (cf. Definition 3).
- $Quality_{ik}$ gives the operation's qualitative properties (cf. Definition 4).

After giving above definitions, we will introduce the semantic description of web services as following:

Definition 6 – *Web service*. A *Web service* WS_i is defined by a tuple $(Description_i, OP_i, Bindings_i, Purpose_i, Category_i)$ where:

- $Description_i$ is a text summary about the service features.
- OP_i is a set of operations provided by WS_i.
- $Bindings_i$ is the set of binding protocols supported WS_i.
- $Purpose_i = \{Purpose_{ik}(op_{ik}) \mid op_{ik} \in OP_i\}$ is a set of WS operations' purpose.
- $Category_i = \{Category_{ik}\ (op_{ik}) \mid op_{ik} \in OP_i\} \cup \{Category_i\ (WS_i)\}$ is a set of WS_i operations' categories.

3.2 Composability model for web services

A major issue in the cooperation of cross-enterprise services is whether they can understand each other and can be composable. For example, it would be difficult for an enterprise service to invoke an operation of other enterprise service if there were no mapping between the parameters (e.g., data types, number of parameters). So, we want to identify two sets of composability rules to compare syntactic and semantic properties of web services (Brahim, 2003).

Syntactic rules include:

1) *mode composability*, which compares operation modes, and
2) *binding composability*, which compares the binding protocols of interacting services.

Semantic rules include:

1) *message composability*, which compares the number of message parameters, their data types, business roles, and units;
2) *operation semantics composability*, which compares the semantics of service operations;
3) *qualitative composability*, which compares qualitative properties of web services.

The detail definitions for the composability rules are introduced as following:

Definition 7 – *Mode composability*. Two operations $op_{ik} = (D_{ik}, M_{ik}, In_{ik}, Out_{ik}, P_{ik}, C_{ik}, Q_{ik})$ and $op_{jl} = (D_{jl}, M_{jl}, In_{jl}, Out_{jl}, P_{jl}, C_{jl}, Q_{jl})$ are *mode composable* if (i) M_{ik} = "notification" and M_{jl} = "one-way"; or (ii) M_{ik} = "one-way" and M_{jl} = "notification"; or (iii) M_{ik} = "solicit-response" and M_{jl} = "request-response"; or (iv) M_{ik} = "request-response" and M_{jl} = "solicit-response".

Definition 8 – *Binding composability*. Two services $WS_i = (D_i, O_i, B_i, P_i, C_i)$ and $WS_j = (D_j, O_j, B_j, P_j, C_j)$ are *binding composable* if $B_i \cap B_j \neq \Phi$.

Definition 9 – *Message composability*. Two operations $op_{ik} = (D_{ik}, M_{ik}, In_{ik}, Out_{ik}, P_{ik}, C_{ik}\ Q_{ik})$ and $op_{jl} = (D_{jl}, M_{jl}, In_{jl}, Out_{jl}, P_{jl}, C_{jl}, Q_{jl})$ are *message composable* if:
(i) $\forall p \in In_{ik}, \exists p' \in In_{jl} \mid p$ is data type compatible with p' and $U(p) = U(p')$, and $R(p) = R(p')$.
(ii) $\forall p \in In_{jl}, \exists p' \in In_{ik} \mid p$ is data type compatible with p', and $U(p) = U(p')$, and $R(p) = R(p')$.

Definition 10 – *Operation semantics composability*. We say that $op_{ik} = (D_{ik}, M_{ik}, In_{ik}, Out_{ik}, P_{ik}, C_{ik}, Q_{ik})$ is *operation semantics composable* with $op_{jl} = (D_{jl}, M_{jl}, In_{jl}, Out_{jl}, P_{jl}, C_{jl}, Q_{jl})$ if (i) P_{ik} is *compatible with* P_{jl} and (ii) C_{ik} is *compatible with* C_{jl}.

Definition 11 – *Qualitative composability*. We say that $op_{ik} = (D_{ik}, M_{ik}, In_{ik}, Out_{ik}, P_{ik}, C_{ik}, Q_{ik})$ is *qualitatively composable with* $op_{jl} = (D_{jl}, M_{jl}, In_{jl}, Out_{jl}, P_{jl}, C_{jl}, Q_{jl})$ if:
(i) Q_{ik}.Fees ≥ Q_{jl}.Fees; and
(ii) (Q_{ik}.Security = true) => (Q_{jl}.Security = true); and
(iii) Q_{ik}.Privacy ⊆ Q_{jl}.Privacy.

Based on the composability model, we propose an efficient model to support the automatic cooperation of cross-enterprise services.
Figure 1 shows the architecture of our model:

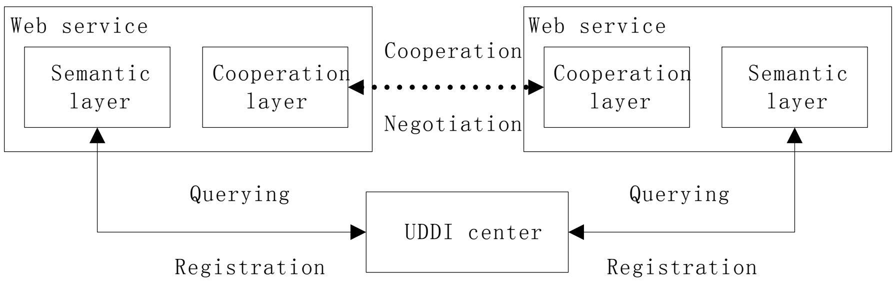

Figure 1. Cooperation model

In this model, there are two layers defined in the web service: one is the semantic layer and the other is cooperation layer. The semantic layer is responsible for defining the semantic rules for the web service. It will register the semantic description of the web service to the UDDI center and query the information of the composable web service for cooperation through the semantic description. Composability rules will be used for UDDI center to find out the composable services. When the composable services are found, cooperation layer will be responsible for the negotiation process to find out the optimum service for cooperation.

4. AUTOMATIC MATCHMAKING OF THE CROSS-ENTERPRISE SERVICES

4.1 Enhanced UDDI center

In order to support the automatic matchmaking among the cross-enterprise services, UDDI center is enhanced to allow the registration of composability rules, which are described in WSDL using XML.
Figure 2 shows the enhanced UDDI center:

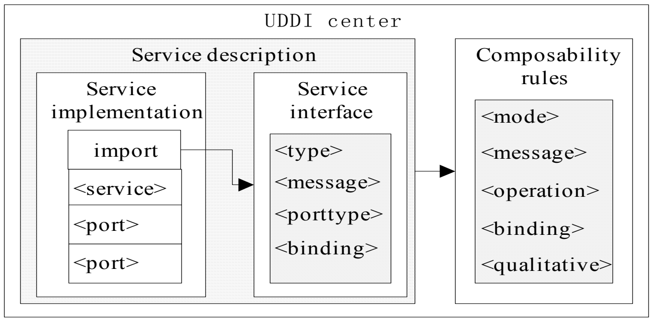

Figure 2. Enhanced UDDI center

Through the enhanced UDDI center, the enhanced WSDL with semantic description is registered for enterprise service to automatically discover the composable cross-enterprise service to achieve the dynamic cooperation. The composability of the cross-enterprise services will be decided according to the synthesis evaluation on the model composability, binding composability, message composability, operation semantics composability and qualitative composability.

4.2 Automatic matchmaking

Figure 3 shows an automatic matchmaking process.

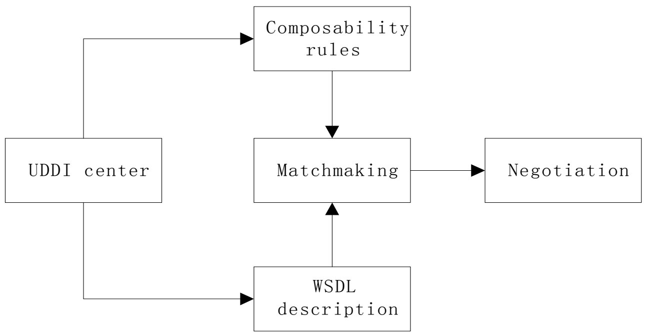

Figure 3. Matchmaking process

There are two ways in the cooperation and composition of cross-enterprise services (Zhou, 2005):

1） Web service internal driven: the cooperation and composition is performed due to the requirement of web service in some enterprise. For example, when the network finance management system in some enterprise wants to implement the functionalities of payoff, tax management and cash management, it should interact with the systems of bank and revenue office to achieve the cooperation.
2） Business requirement driven: the cooperation is established due to the business requirement advanced by some enterprise through the integration of several business systems (web services). For example, when some enterprise wants to invest on some fund, it should obtain the invest

information of the fund such as stock, exchange and bond, etc. So it requires other organizations (i.e. stock exchange, bank) to be integrated to provide the related analysis information on the fund.

For the composition and cooperation of the cross-enterprise services, a composition plan should be built first according to the business logic and composability rules whatever it is web service internal driven or business requirement driven. By composition plan, we mean the list of component services and their interactions with each other (plugging operations, mapping messages, etc.) to form the composite service. The matchmaking algorithm uses as input the composer's specification and UDDI of preexisting service interfaces described in WSDL (extended with semantic constructs). Through the composition plan, a detailed description of the composite service is automatically generated. This description includes the list of outsourced services, mappings between the composite and outsourced services operations and messages, and the control flow of outsourced operations. The control flow refers to the execution order of the operations outsourced by the composite service.

After the component services are automatically matched and selected, the negotiation process will be adopted to achieve the win-win cooperation among the cross-enterprise services.

5. POLICY DRIVEN NEGOTIATION

Some composable services will be chosen out as candidate services to attend the policy driven negotiation process. The policy is described using XML, which are defined in the extended WSDL. Through the multi-steps negotiation process driven by policies, the final cooperation of the cross-enterprise services will be achieved.

5.1. Negotiation Policy in XML

Policies (Yang, 2002; Tang, 2005) are rules governing the choices in the behavior of a system. In our model, policy is adopted to address the adaptive negotiation process.

The negotiation policies are expressed in XML since it has a text-based representation which imposes few restrictions on network technology or protocols and (through the use of XML schema) it has a sufficiently strict syntax to permit automated validation and processing of information in an unambiguous way.

Negotiation policies allow selections from a range of options provided by the developer of the policy-controlled service, comprised of rules in the form of if<condition>then<action> (Tang, 2005). This allows flexibility to be build into a system by supporting a range of different behaviors rather than hard-coding a particular behavior. Furthermore, the use of XML as an intermediate representation still allows the negotiation policies to be developed using any existing approach. Some kind of policy editor GUI can be provided to give a user-friendly and flexible way to manage the service-specific policies.

The following shows the information details, which will be included in the negotiation policy (Figure 4 shows an example of a service policy in XML):

- **Service:** the referenced information of the service such as service id, service name and service type, etc;

- **Info:** the elementary information on the policy itself including policy id, creator, etc. The policy id and service id will be used to identify the service policy;
- **Param-specs:** the available parameters (information) for evaluating the conditions, such as cost, profit ratio, bottom price, QoS, minimal/max adjusted value, etc;
- **EvalFunction:** the evaluation function of synthetic parameters for decision making on actions under different conditions;
- **Trigger:** the relevant policies triggered by different results from the evaluation function to trigger the actions;
- **Action:** the details of actions on the evaluation of the local available parameters.

```
<?xml version = "1.0" encoding = "UTF-8"?>
<policy xmlns = " http://~ tc/policy" xmlns:xsi =
"http:// ~tc /XMLSchema-instance" xsi:schemaLocation
="http:// ~tc /policy file:///home/~tc /policy.xsd">
<service>
    <service-id>SIN001</service-id>
    <name>service001</name>
    <type>Single</type>
</service>
<info>
    <policy-id>Policy1</policy-id>
    <creator>tjf</creator>
    <authority>ALL</authority>
</info>
<param-specs>
    <param-spec variable=var_1>
        <description>descript_1</description>
    </param-spec>
    < param -spec variable=var_2>
        <description>descript_1</description>
    </param -spec>
</param-specs>
<evalFunction value=evalfun_variable>
    EvalFun(param-specs)
</evalFunction>
<trigger variable=evalfun_variable>
    <Branch>
        <Condition value=value_1>
                <Action>Act_1</Action>
        </Condition>
        <Condition value=value_2>
                <Action>Act_2</Action>
        </Condition>
    </Branch>
</trigger>
</policy>
```

Figure 4. An example of negotiation policy in XML

5.2. Negotiation process

The specific requirements will be presented by composer for the candidate services, such as QoS, fee, etc. The candidate services will perform evaluation on the requirements according to the negotiation policy and provide the corresponding bids with some contact. During the negotiation process, some adjustments are allowed to be done on the contact, which is also according to the negotiation policy. When the negotiation process ends, the composer will evaluate the final bids and choose the optimum service to establish the cooperation. Figure 5 shows the policy driven negotiation process.

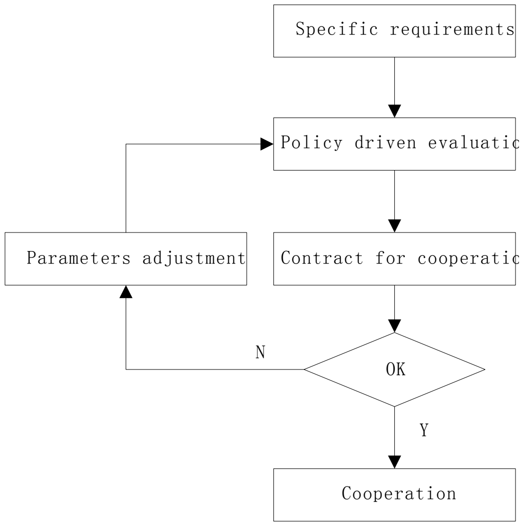

Figure 5. Policy driven negotiation process

Through such policy driven negotiation process, the cross-enterprise services can achieve win-win among the dynamic cooperation process.

6. CONCLUSION

This paper has introduced a service composability model to support dynamic cooperation of cross-enterprise services by applying the semantic rules of composability description. An enhanced UDDI center is provided to support the registration of composability rules by web services. There are two layers defined in web service: semantic layer and cooperation layer. Semantic layer is responsible for the automatic matchmaking process to select the composable services, while cooperation layer is responsible for the negotiation process to establish the final cooperation. Such policy driven negotiation process can achieve win-win among the dynamic cooperation of cross-enterprise services.

7. REFERENCES

1. A. Berfield, P. Chrysanthis, I. Tsamardinos, M. Pollack, S. Banerjee, “A Scheme for Integrating E-Services in Establishing Virtual Enterprises”, Proceedings of Research Issues in Data Engineering (RIDE) Workshop, San Jose, CA, February 2002.
2. T. Berners-Lee, J. Hendler, O, Lassila. The Semantic Web. Scientific American 284(5), pp. 34-43. May 2001.
3. B. Medjahed, A. Bouguettaya, A.K. Elmagarmid, “Composing Web Services on the Semantic Web”, The VLDB Journal, 12(4), November 2003.
4. C. Bussler, D. Fensel, A. Maedche. A conceptual architecture for Semantic Web enabled Web services. SIGMOD Record 31(4): 24–29, 2002.
5. I. Horrocks. DAML+OIL: A description logic for the Semantic Web. IEEE Data Engineering Bulletin 25(1): 4–9. March 2002.
6. S.A. McIlraith, T.C. Son, H. Zeng. Semantic Web services. IEEE Intelligent Systems. Special Issue on the Semantic Web. 16(2): 46–53, March/April, 2001.
7. J.F. Tang, B. Zhou, Z.J. He, P. Uros, “Adaptive Workflow-Oriented Services Composition and Allocation in Distributed Environment”, Proceedings of the third International Conference on Machine Learning and Cybernetics, Shanghai, China, August 2004, pp.599-603.
8. J.F. Tang, B. Zhou, Z.J. He, “Dynamic Economic- and QoS-Based Approach for Workflow-Oriented Distributed Services Allocation”, Proceedings of the International Conference on Systems, Man and Cybernetics, Hague, Netherlands, October 2004, pp. 3349-3353.
9. J.F. Tang, B. Zhou, Z.J. He, “Policy driven and multi-agent based fault tolerance for Web services”, Journal of Zhejiang University SCIENCE, 2005 6A (7): 676-682.
10. V. Ouzounis, V. Tschammer, “A Framework for Virtual Enterprise Support Services”, Proceedings of the 32nd Hawaii International Conference on System Sciences, 1999.
11. G. Weikum (editor). Special issue on Organizing and Discovering the Semantic Web. IEEE Data Engineering Bulletin 25(1): 1–58. March 2002
12. J. Yang, W.J. van den Heuvel and M.P. Papazoglou, "Service Deployment for Virtual Enterprises, Proceedings of the Workshop on Information Technology for VE, 2001.
13. K. Yang, A. Gailis, Todd, C., 2002. Policy-Based Active Grid Management Architecture. Proceedings of 10th IEEE International Conference on Networks (ICON02), p.243-248.
14. B. Zhou, J.F. Tang, Z.J. He, “An adaptive model of virtual enterprise based on dynamic web service composition”, Proceedings of the Fifth International Conference on Computer and Information Technology, September 2005, pp. 284-289.