A Business-Level Service Model Supporting End-User Customization*

Jianwu Wang[1,2] and Jian Yu[1]

[1] Software Engineering Research Group, Dept. of Control and Computer Engineering, Politecnico di Torino, 10129, Torino, Italy
[2] Research Centre for Grid and Service Computing, Institute of Computing Technology, Chinese Academy of Sciences, 100080, Beijing, China
{jianwu.wang,jian.yu}@polito.it

Abstract. If end users can utilize the abundant Web services to construct business applications on demand and independently, the ever changing business requirements will be met efficiently and timely. However, Web services are difficult for end users to use directly due to both the diversity of Web services and the diversity of end-user requirements. To tackle the problem, we introduce feature modeling and configuring techniques in domain engineering into service-oriented computing, and correspondingly propose a business-level service model and an end-user friendly service customization mechanism. Feasibility of the proposals is demonstrated on an example.

Keywords: Domain-Specific Reuse, Business-Level Service Model, Service Customization, End-User Friendly, Variability-Supported Service Matching.

1 Introduction

Nowadays, business requirements are ever changing in many business domains, such as scientific research, government and commence, which requires information systems to be integrated on demand to handle the requirements in a just-in-time manner. With the gradual popularity of service-oriented computing technologies, there are more and more Web services available on the Internet. In the bioinformatics domain, for instance, the number of Web services has added up to over 3000 [1]. If end users can utilize these services to construct business applications on demand and independently, the ever changing business requirements will be met efficiently and timely. However, Web services are difficult for end users to use directly due to both the diversity of Web services and the diversity of end-user requirements.

To alleviate the above problem, we propose a business-level service approach (named *VINCA Business Service Approach*) based on our previous work [2, 3]. The approach consists of a business-level service model (named *VINCA Business Service Model*) with end-user friendliness and domain-specific reusability, and a corresponding service

* This research was supported by National Natural Science Foundation of China (No. 60573117).

E. Di Nitto and M. Ripeanu (Eds.): ICSOC 2007 Workshops, LNCS 4907, pp. 295–303, 2009.

modeling and reusing mechanism. This approach can facilitate end users to construct on demand business applications timely by themselves.

The service model and the corresponding service customization mechanism will be discussed in this paper. The rest of the paper is organized as follows. In Section 2, we present a motivating example. In Section 3, the service model is explored in detail. Then, the corresponding service customization mechanism is discussed in Section 4. Section 5 compares with related work. Finally, we draw a conclusion and briefly state our future work.

2 Motivating Example

We will use a simplified example from the weather service domain throughout the paper (see Fig. 1). There are over 15 Web services (including 179 independent operations)[2] providing weather forecast on the Internet. When the number of Web services with similar functionality is huge, it is very difficult for end users to directly select proper services and reuse them.

We can then split the problem into two parts:

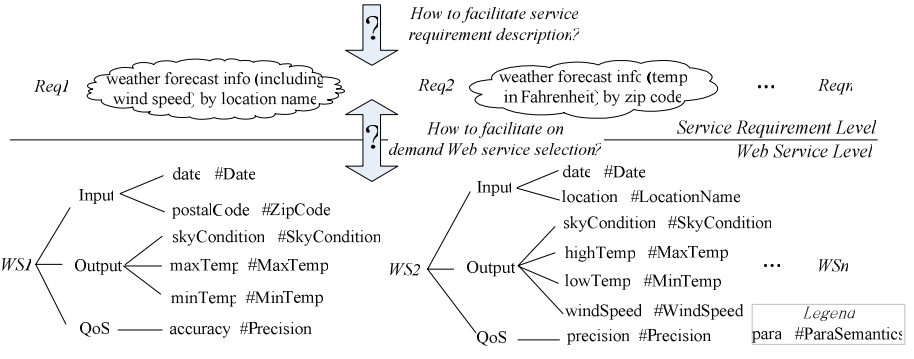

Fig. 1. Two levels of service usage in service-oriented environments

1) *Similarity and diversity of end-user requirements*: As shown in Fig.1, *Req1* and *Req2* are two similar yet different service requirements of end users. For instance, wind speed information is mandatory in *Req1* but not in *Req2*; the preferred ways to describe location are also different. A key problem at the service requirement level is how to facilitate end users to describe their requirements in a certain business domain where end-user requirements are similar yet diverse.

2) *Similarity and diversity of Web services*: Also as shown in Fig.1, *WS1* and *WS2* are two similar yet different Web services. For instance, their input parameters for location are different; moreover, *WS2* has an additional output: wind speed. A key problem at the Web service level is how to optimize the matching between service requirements and executable Web services in a service-oriented environment where

[2] An incomplete list can be found at from http://softeng.polito.it/wang/domainservice/WeatherForecastWSList.html.

Web services are abundant yet diverse in capability (namely Input, Output, Precondition, Effect and QoS).

3 VINCA Business Service Model

To tackle the above problems, we propose VINCA business service approach (Fig. 2). As the core of the approach, VINCA business service model (or business service for short) will be explained in detail through the following three subsections: principle, components, and matching with Web services.

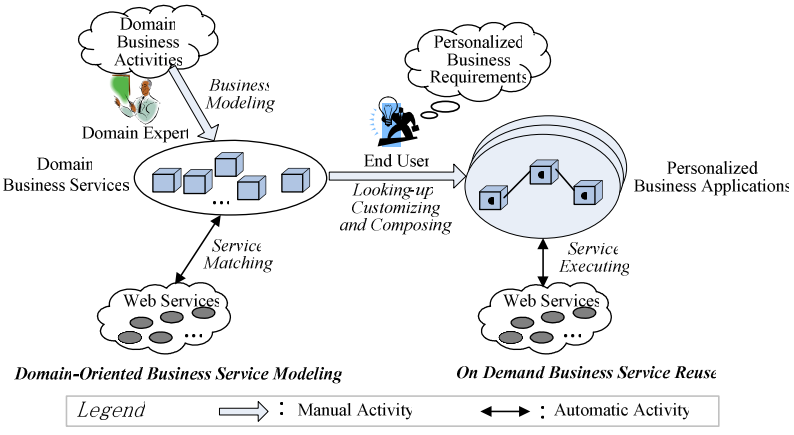

Fig. 2. The overview of VINCA business service approach

3.1 The Principle of VINCA Business Service Model

The principle of business service model can be summarized into two main aspects:

End-User Friendliness: Web services are usually utilized by end users to perform certain domain-specific business activities [4]. So each business service is modeled as an independent and executable business activity by combining top-down domain-specific activity modeling and bottom-up Web service abstraction. From real project experiences, we summarize business activity requirements into four typical categories: business activity looking-up, customization, composition and execution. To meet these typical requirements, business service is modeled in a layered fashion, so that end users can utilize business services through unfolding the corresponding layer according to the type of their requirements.

Domain-Specific Reusability: Domain-specific reuse is an efficient reuse mechanism which has been successfully applied in domain engineering [5, 6]. One of its core mechanisms is modeling commonalities and variabilities within a certain domain. In service-oriented environments, the commonalities and variabilities are represented as similar yet diverse business activity requirements (a.k.a. service requirements) at the business level, and similar yet diverse Web services at the IT

level (see Fig.1). So they are modeled at both the business level and the IT level in business service model. Then we can get domain-specific reusable business services (called *domain business services*) bound with proper Web services, which can facilitate end users to describe personalized requirements and utilize bound Web services transparently.

3.2 The Components of VINCA Business Service Model

Following the thought of "separation of concerns", business service is modeled as layers and formed an *iceberg* (shown in Fig. 2.). Firstly, business related and IT related information of the model is separated. Secondly, business related information is separated further according to its functionalities. The upper three layers depict business related information, which are presented to end users and correspond to the business activity looking-up, customization and composition requirements respectively. The bottom layer depicts IT related information, which is presented to IT professionals and corresponds to the business activity execution requirement.

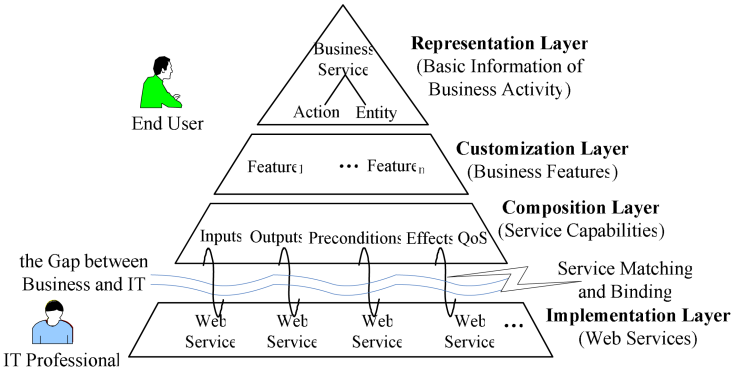

Fig. 3. The iceberg model of VINCA business service

- Representation Layer

This layer depicts the basic function of business activities, which is the basic information for end users to choose business services. Each business activity can be simply represented as two basic parts: action and entity (namely object of action) [7, 8]. So we abstract business activities as combinations of action and entity in the representation layer which is used as the fundamental information to depict and differentiate business activities. Business service names also usually adopt a form of "action + entity" (such as Enquiry Route) or "entity + action" (such as Weather Forecast).

- Customization Layer

This layer depicts the detailed business properties of business activities, which is for end users to describe their personalized requirements. Feature modeling techniques has been widely applied in domain engineering to depict business properties of applications within a certain domain [7, 8]. From feature models, end users will know whether an application can meet their requirements, and describe their personalized

requirements through feature configuration. So we employ feature modeling techniques in the customization layer, which can facilitate end users to know the detailed business properties of business activities and describe their personalized business activity requirements.

- Composition Layer

This layer depicts the capability information of business activities, which is for end users to compose services at the business level [2, 9]. Referring to the way of OWL-S semantic Web service model [10], the composition layer depicts service capabilities as input, output, precondition, effect and QoS. Moreover, the capability information can also be used to match Web services.

- Implementation Layer

This layer depicts the information of bound Web services that perform business activities. Then Web services can be selected and invoked when the business service is executed. Concretely, the WSDL file information of bound Web services is recorded in the implementation layer.

It is worth noting that the information in the customization layer and the composition layer can be seen as the same information depicted from two perspectives: business property and capability. On one hand, business services interact with end users through their business properties; on the other hand, business services match and bind Web services through their capabilities. So the business property and capability can be correlated to realize the selection and invocation of Web services according to end-user requirements. We depict the correlation by specifying these features' functionalities in capability, namely which features in the customization layer will be used in input, output, precondition, effect and QoS in the composition layer.

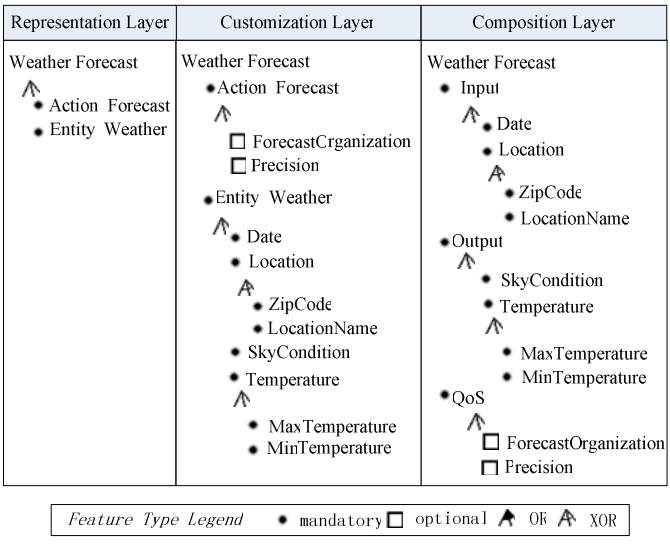

Fig. 4. Business related information of *weatherforecast* business service

For the examples and problems presented in Section 2, a domain business service in the weather service domain, *WeatherForecast*, is modeled, whose business related information is shown in Fig. 4 (precondition and effect of the service is omitted here). Since the weather location is typically described as zip code or location name, but not both, so they are modeled as two sub features of location, and the feature type is XOR. Wind speed is an additional property for typical weather forecast, so it is not included in *WeatherForecast*.

Following the above mechanism, the commonalities and variabilities of services can be modeled by feature modeling techniques in both the customization layer and the composition layer. Then business service model can adapt to the diversity of end user requirements and the capability diversity of Web services respectively. The commonalities and variabilities modeled in the customization layer can facilitate end users to understand and customize services, and those modeled in the composition layer can support the matching with Web services that may have diverse capabilities.

3.3 Matching between Business Services and Web Services

Web services are needed for business service execution, so a semantic and variability-supported service matching mechanism is proposed to realize automatically binding between one business service and proper Web services with diverse capabilities. The mechanism has the following two advantages compared to related semantic service matching mechanisms [10, 11, 12]. Concrete service matching mechanism is omitted for the space limitation.

1) *Variability Supported*: Feature modeling techniques are adopted in business service model so that each business service may has many possible capabilities. If a Web service can match one possible capability of a certain business service, it can be bound to the business service.

2) *Additional Property Allowed*: Business services represent typical domain specific business activities, but Web services are provided independently and then may have additional business properties. For example, typical properties of weather forecast only include sky condition, max temperature and min temperature, but independent Web services may have additional properties, such as wind speed. To adapt to this characteristic, Web services with additional business properties can also be bound to the corresponding business services.

For the example of weather forecast, the capability information of *WS1* and *WS2* belongs to the possible capabilities of *WeatherForecast*, so both of them can match *WeatherForecast*. Then the capability diversity of underlying Web services is hidden from end users. Moreover, the business service information that presented to end users is independent of the changes of Web services.

4 End-User Friendly Service Customization

As shown in Fig. 2, domain business services are modeled by domain experts in advance for reuse. Yet there are always slight differences between pre-modeled business services with personalized end-user requirements. Since end users know their requirements well, so we present a service customization mechanism to facilitate end

users to describe their personalized requirements by themselves on the base of reusing proper domain business services. Note that the service customization mechanism is only to meet single service requirements; complex requirements can be met through the composition of customized services, which is beyond the scope of this paper.

Three categories of customization operations are proposed for end users to customize:

- Adding New Features

If an end-user requirement has more business property demands than the business properties of a certain domain business service, the end user can specify this requirement by adding additional features in the customization layer of the business service, such as adding a new mandatory sub feature to an existing feature.

- Deleting Existing Features

If an end-user requirement only has partial business properties of a certain domain business service, the end user can specify this requirement by deleting additional features in the customization layer of the business service, such as deleting an existing mandatory sub feature of an existing feature.

- Configuring Existing Features

If an end-user requirement belongs to the business property variability (namely the range of possible business properties) of a certain domain business service, the end user can specify this requirement by configuring features in the customization layer of the business service, such as choosing one or more sub features of an OR feature.

To perform personalized requirements of end users, proper Web services should be selected according to the service customization results. While end users customize a business service in its customization layer, its composition layer is also changed correspondingly through the correlation of the two layers. With the service matching mechanism presented in Section 3, the capability information in the composition layer can be used to match and get proper Web services meeting the end-user requirements. The above customization operations only involve business related information of business services, so they can be easily used by end users.

For the example of weather forecast, end users can describe their personalized weather forecast requirements by customizing the domain business service: *WeatherForecast*. For *Req1* in Fig.1, the end user need to choose the location name sub feature of the location feature, and add wind speed as a new feature of weather in the customization layer of *WeatherForecast*. In the composition layer of *WeatherForecast*, the location input parameter will be automatically specified as location name, and wind speed feature should be manually specified as an output parameter for it is a new feature. Of *WS1* and *WS2*, only *WS2* can match the customization result according to the matching mechanism in Section 3. Then *WS2* can be executed to satisfy the personalized requirement.

5 Related Work

In this section, we will discuss two areas of related work: service model and service customization.

Service Model: Many service models (and service matching mechanisms) have been proposed to abstract Web services with similar capabilities and then simplify service

usage [10, 11, 12, 13]. But they consider little about the capability diversity of dynamic and independent Web services. So either there are too many abstract services for end users to select and use, or there are many Web services that can not automatically match abstract services. In VINCA business service model, the commonalities and variabilities of services are modeled, then not only the number of business services can be kept in a minor range, but also most of Web services can match business services.

Service Customization: Recently, some effort has been put into importing feature modeling in service-oriented computing. In [14], each feature represents a service operation, which can support operation variabilities in similar systems. Feature modeling is also used to express non-functional properties of services [15, 16] and implement techniques [17]. Customization on these aspects can be realized through feature configuration, yet they do not discuss the concrete customization mechanism, such as customization operations. Moreover, none of these works deal with the service capability variability, which is a main difficulty for end users to directly select Web services. In VINCA business service model, features are used to depict business properties and capabilities of services, and three categories of customization operations are provided. Corresponding service matching mechanism is also proposed to select suitable Web services according to the customization results.

6 Conclusion and Future Work

Service abstraction is necessary and valuable to facilitate on demand Web service usage. Hereinto, the diversity of similar Web services and the diversity of similar service requirements are two important factors need to be considered. In this paper, feature modeling and configuring techniques in domain engineering are introduced into service model and customization to adapt to the diversity of similar service requirements and the capability diversity of similar Web services, which obtain a business-level service model and an end-user friendly service customization mechanism.

For future work, since the diversity of real Web services and service requirements is very complicated, we are extending our model to have more expressive power, such as feature constraints. Moreover, more and in-depth empirical experiments will be made to obtain evidence, which can testify the advantages of our work.

References

1. Hull, D., Zolin, E., et al.: Deciding Semantic Matching of Stateless Services. In: 21st National Conference on Artificial Intelligence and 18th Innovative Applications of Artificial Intelligence Conference (AAAI 2006), pp. 1319–1324 (2006)
2. Han, Y., Geng, H., Li, H., Xiong, J., Li, G., Holtkamp, B., Gartmann, R., Wagner, R., Weißenberg, N.: VINCA – A visual and personalized business-level composition language for chaining web-based services. In: Orlowska, M.E., Weerawarana, S., Papazoglou, M.P., Yang, J. (eds.) ICSOC 2003. LNCS, vol. 2910, pp. 165–177. Springer, Heidelberg (2003)

3. Wang, J., Yu, J., et al.: A Service Modeling Approach with Business-Level Reusability and Extensibility. In: 1st IEEE International Workshop on Service-Oriented System Engineering (SOSE 2005), pp. 23–28 (2005)
4. Baida, Z., Gordijn, J., et al.: A Shared Service Terminology for Online Service Provisioning. In: 6th International Conference on Electronic Commerce (ICEC 2004), pp. 1–10 (2004)
5. Kang, K.C., Cohen, S.G., et al.: Feature-Oriented Domain Analysis Feasibility Study. Technical Report: SEI-90-TR-21. Pittsburgh, Software Engineering Institute, Carnegie Mellon University (1990)
6. Czarnecki, K., Eisenecker, U.W.: Generative Programming: Methods, Tools and Applications. Addison-Wesley, New York (2000)
7. Arisha, K., Kraus, S., et al.: Impact: A Platform for Collaborating Agents. IEEE Intelligent Systems 14(2), 64–72 (1999)
8. Ikeda, M., Seta, K., et al.: Task Ontology: Ontology for Building Conceptual Problem Solving Models. In: 13th European Conference on Artificial Intelligence (ECAI 1998), Brighton, England (1998)
9. Yu, J., Wang, J., et al.: Developing End-User Programmable Service-Oriented Applications with VINCA. In: Workshop on Web Logistics (2004)
10. Martin, D., Paolucci, M., McIlraith, S.A., Burstein, M., McDermott, D., McGuinness, D.L., Parsia, B., Payne, T.R., Sabou, M., Solanki, M., Srinivasan, N., Sycara, K.P.: Bringing semantics to web services: The OWL-S approach. In: Cardoso, J., Sheth, A.P. (eds.) SWSWPC 2004. LNCS, vol. 3387, pp. 26–42. Springer, Heidelberg (2005)
11. Paolucci, M., Kawamura, T., Payne, T.R., Sycara, K.P.: Semantic matching of web services capabilities. In: Horrocks, I., Hendler, J. (eds.) ISWC 2002. LNCS, vol. 2342, pp. 333–347. Springer, Heidelberg (2002)
12. Tan, Y., Vellanki, V., et al.: Service Domains. IBM Sytems Journal 43(4), 734–755 (2004)
13. Benatallah, B., Sheng, Q., et al.: The Self-Serv Environment for Web Services Composition. IEEE Internet Computing 7(1), 40–48 (2003)
14. Chen, F., Li, S., et al.: Feature Analysis for Service-Oriented Reengineering. In: 12th Asia-Pacific Software Engineering Conference (APSEC 2005), pp. 201–208 (2005)
15. Wada, H., Suzuki, J., et al.: A Feature Modeling Support for Non-Functional Constraints in Service Oriented Architecture. In: 2007 IEEE Int. Conf. on Services Computing (SCC 2007), pp. 187–195 (2007)
16. Fantinato, M., de S. Gimenes, I.M., de Toledo, M.B.F.: Supporting QoS Negotiation with Feature Modeling. In: Krämer, B.J., Lin, K.-J., Narasimhan, P. (eds.) ICSOC 2007. LNCS, vol. 4749, pp. 429–434. Springer, Heidelberg (2007)
17. Robak, S., Franczyk, B.: Modeling web services variability with feature diagrams. In: Chaudhri, A.B., Jeckle, M., Rahm, E., Unland, R. (eds.) NODe-WS 2002. LNCS, vol. 2593, pp. 120–128. Springer, Heidelberg (2003)