A Feature-Oriented Approach for Web Service Customization

Tuan Nguyen
Centre for Complex Software Systems & Services
Swinburne University of Technology, Melbourne, Australia
E-mail: tmnguyen@swin.edu.au

***Abstract*—This paper presents our ongoing work of a feature-oriented approach for Web services customization that helps to reduce the complexity in the customization process and enables the automated validation of customized services. Specifically, it exploits the feature modeling techniques of Software Product Line (SPL) to capture all possible customization options into a feature model. Weaving models, describing the links between the feature model and service models, enable service consumers to customize services through the selection of features in the feature model. By representing the capability of Web services at higher level of abstraction, i.e. feature level, it helps to simplify the customization process so that service consumers can focus on what features they want, rather than the technical details (e.g. operations, messages) of how to achieve it. In addition, automated analysis of feature models enables the automated validation of customized services.**

Keywords: Service customization, feature model, variability modeling.

I. INTRODUCTION

An increasing number of organizations are turning to Service-Oriented Architectures (SOAs), based on Web service technologies, to consolidate and repurpose legacy applications, and combine them with new applications. Along with the proliferation of Web services, service ecosystems[1] are emerging, in which service providers interconnect their offerings in (un)foreseen ways to provide customized, value-added services. Services in Web service ecosystems have to satisfy various consumer demands. It is not likely that all consumers of a specific service have the same set of requirements on the service offered. Rather, such requirements are slightly different from one consumer to another. Therefore, supporting service customization is a crucial requirement in service ecosystems.

Service customization refers to operations of adapting a service to a particular application scenario. This is a non-trivial task that requires both technical knowledge and business expertise [2]. The challenges for an efficient service customization framework come from two perspectives. Firstly, actual business services support a plethora of possible customization options with a massive numbers of dependencies scattered between those options. The customization framework needs to efficiently address the complexity of the customization process. Secondly, due to previously explained complexity, service customization is a very error-prone process. The customization framework has to efficiently support the validation of customized services.

Software Product Line (SPL)[3] is a software engineering paradigm aiming at developing a family of software systems (or products) from reusable core assets. The key success of SPL is the use of feature models to model and manage variability in the product family. We argue that an efficient variability modeling technique is sufficient to address the challenges of service customization. In this paper, we propose an approach exploiting the feature modeling techniques of SPL to support service customization. Comparing to related works, our approach is more advantageous in term of supporting both reduced complexity and automated validation.

The paper is structured as follows. Section II describes our customization framework, its advantages, as well as the detail of how to realize it. We present related works in section III. We then conclude the paper and outline the future works in section IV.

II. SERVICE CUSTOMIZATION FRAMEWORK

A. Feature modeling technique and its benefits

In Software Product Line (SPL), a feature is defined as a visible characteristic of the product family [4]. A feature model represents the information of all possible products of a software product line in terms of features and relationships among them. A feature model is represented as a hierarchically arranged set of features composed by:

1. Relationships between a parent feature and its child features. Possible relationships are *mandatory, optional and alternative*.
2. Cross–tree (or cross–hierarchy) constraints (a.k.a. feature dependencies) that are typically inclusion or exclusion statements of the form: *if feature F is included, then features A and B must also be included (or excluded).*

Figure 1 depicts a simplified feature model representing a family of Video on Demand (VoD) services. According to the model, all customized VoD services must support *Streaming*, *Device* and *Payment* features. *Messaging* is optional feature that can be selected by a particular consumer. Consumers can select between high resolution (i.e. *HighRes*) and low resolution (i.e. *LowRes*), PC-oriented service (i.e. *PC*) and mobile-oriented service (i.e. *MobilePhone*), as well as payment by credit card (i.e. *CreditCard*) or through mobile carrier (i.e. *MobileCarrier)*. In addition, because of feature dependencies, if a consumer selects mobile-oriented service, he has to use low resolution and payment by mobile carrier.

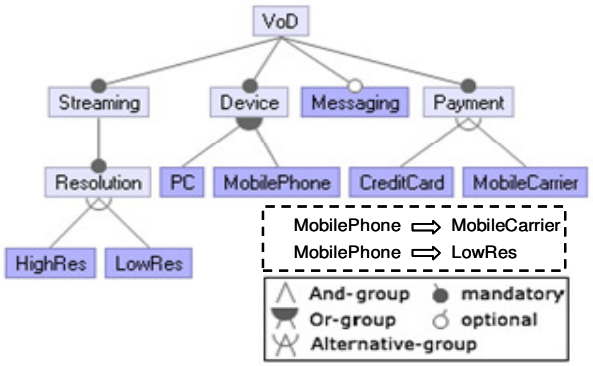

Figure 1. A sample feature model

Feature is an effective communication "medium" among different stakeholders. It is often the case that business analysts and engineers speak of service characteristics in terms of "features the service has and/or delivers." Therefore, it is very intuitive to enable service customization based on feature models. More importantly, feature models enable modeling variability at the feature level, which generally is higher level of abstraction than implementation details (e.g. variability in operations or messages of service interfaces). Therefore, feature models greatly reduce the number of variation points available for customization. Consequently, it helps to reduce the complexity of service customization.

Moreover, researches in feature modeling techniques have achieved great advance in term of techniques for automated analysis [5]. The automated analysis of feature models is about extracting information from feature models using automated mechanisms. Examples of such extraction include but not limited to: deciding whether a feature configuration (i.e. feature selection) is valid or not; counting the number of possible products; applying filter to derive a subset of possible products; etc. Such automated analysis will help to automatically validate a customization.

B. *Overview*

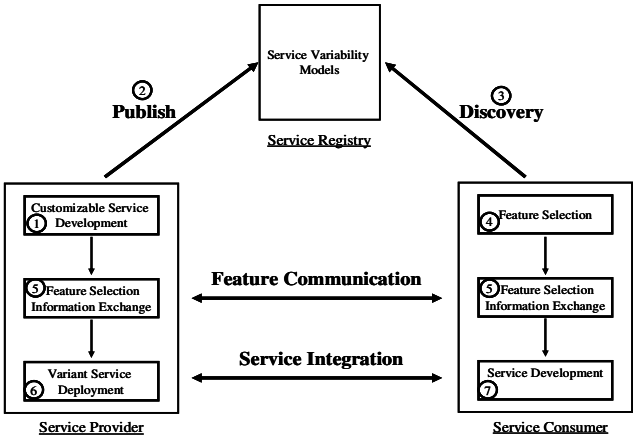

Figure 2. Service Customization Process

In this section, we describe the overall steps of our service customization framework (Figure 2). At the first step, a service provider develops a customizable service as a product line. That is, the customizable service is developed as a family of services such that each family member is a possible customized service for a particular application scenario. To this end, the service provider captures the commonalities and variabilities of the service family into a feature model. The feature model will be used as the requirement model for developing service interface and service implementation. The detail of how to develop service interface and service implementation based on feature models will be explained later.

At the second step, the feature model is published to service registries as a part of service descriptions so that interested consumers can search for it. This step is different from traditional SOA in the term that the service variability model, i.e. the feature model, rather than the service interface description is published to service registries. The reason is, a service interface that a particular consumer consumes will be the result of a runtime customization process and it is not finalized at the time service is published to registries. At the third step, a service consumer discovers the feature model of interested services that it can customize.

The customization process at the consumer's side starts from step 4. In this step, the service consumer selects desired features from the feature model. Feature selection operations include resolving variation points by enabling/disabling optional features and selecting a particular feature from an alternative feature group. Tooling support will help to simplify this process such that it will automatically resolve feature dependencies to prevent invalid feature selections or provide graphical interface for easy customization operations. The result of this step is a particular feature configuration.

Feature selection information is exchanged between the service consumer and the service provider (i.e. step 5) so that the service provider knows exactly which features the consumer desires. Based on this information, the service provider generates a particular service interface as well as deploys a particular service instance binding with that service interface (i.e. step 6). As the result of this, a particular WSDL description is returned to the consumer. At step 7, the service consumer will develop their own services integrating the customized service provided by the service provider.

C. *Feature-oriented service development*

In this section, we describe a model-driven approach to develop a customizable service conforming to a feature model (i.e. step 1 of our framework). The service engineering process is illustrated in Figure 3. Firstly, service developers model service capabilities to generate a feature model. Service capabilities in all possible application scenarios are structured into the feature model in which variation points represent customization options. Then, the service model template and process model template are developed using a superimposed variants approach of Software Product Line [6]. It is based on the idea of creating a *model template* containing all family members in a superimposed way. The specialization of such a template gives rise to different members (i.e. customized services or *template instance*) and it is carried out by purging model

template following the selection of features from a feature model, i.e. a feature configuration. Model template is itself a model expressed in the same notation as the template instance. Therefore, in our framework, service model template will be described using a service modeling language such as SoaML [7]. Likewise, process model template will be described using a process modeling language such as BPEL.

The superimposed variant technique requires mechanisms to relate features from the feature model to model template. In our approach, we use weaving models [8] for this purpose. Weaving models describe links between the feature model and other two models, i.e. service model template and process model template. Such weaving models help to capture relationships between model elements which will be used to generate a particular customized service.

III. RELATED WORKS

Most existing approaches for service customization is limited to low-level technical aspects, such as the configuration of technical parameters for invocation and runtime. For such approaches, reducing complexity and automated validation remains grand challenges. [9] presents a policy-driven approach for service customization. However, this approach works with mostly informal policy descriptions that do not allow an automated validation of customization decisions. [2] also exploits variability modeling techniques from SPL to support service customization. Although the approach in [2] enables automated validation of customized services, variability is still modeled at low level of abstraction, i.e. addressing variability in operations, messages of service interface. Therefore, the approach can not address the complexity issue. Our approach is the only one that can address both challenges of service customization. Moreover, the approach also articulates the development process for service implementation conforming to the customizable service interface. This feature is also lack in both [2] and [9].

IV. CONCLUSIONS AND FUTURE WORK

In this paper, we describe a feature-oriented approach for web service customization. Feature modeling techniques of SPL is exploited to address two grand challenges of service customization, i.e. reducing complexity and automated validation. The key concept is to use a feature model as the basis of service customization and utilizing weaving models to reflect customization decisions, i.e. feature configuration, to service interface and service implementation. We are going to develop a prototypical system to evaluate the feasibility of our approach.

REFERENCES

1. Barros, A.P. and M. Dumas, *The Rise of Web Service Ecosystems.* IT Professional, 2006. **8**(5): p. 31-37.
2. Stollberg, M. and M. Muth. *Service Customization by Variability Modeling*. in *Proceedings of Fifth International Workshop on Engineering Service-Oriented Applications - ICSOC 2009 Workshops*. 2009.
3. Pohl, K., et al., *Software Product Line Engineering: Foundations, Principles and Techniques*. 2005: Springer-Verlag New York, Inc.
4. Lee, K., K. Kang, and J. Lee, *Concepts and Guidelines of Feature Modeling for Product Line Software Engineering*, in *Software Reuse: Methods, Techniques, and Tools*. 2002. p. 62-77.
5. Benavides, D., S. Segura, and A. Ruiz-Cortés, *Automated Analysis of Feature Models: A Detailed Literature Review. Version 1.0.* Dec 2009, ISA Research Group, University of Sevilla.
6. Czarnecki, K. and M. Antkiewicz, *Mapping Features to Models: A Template Approach Based on Superimposed Variants*, in *Generative Programming and Component Engineering*. 2005. p. 422-437.
7. Berre, A.J., *Service Oriented Architecture Modeling Language (SoaML) - Specification for the UML Profile and Metamodel for Services (UPMS).* 2008-11-01, OMG.
8. Didonet, M., et al. *Weaving Models with the Eclipse AMW plugin*. in *Proceedings of Eclipse Modeling Symposium, Eclipse Summit Europe*. 2006.
9. Liang, H., et al., *A Policy Framework for Collaborative Web Service Customization*, in *Proceedings of the Second IEEE International Symposium on Service-Oriented System Engineering*. 2006, IEEE Computer Society.

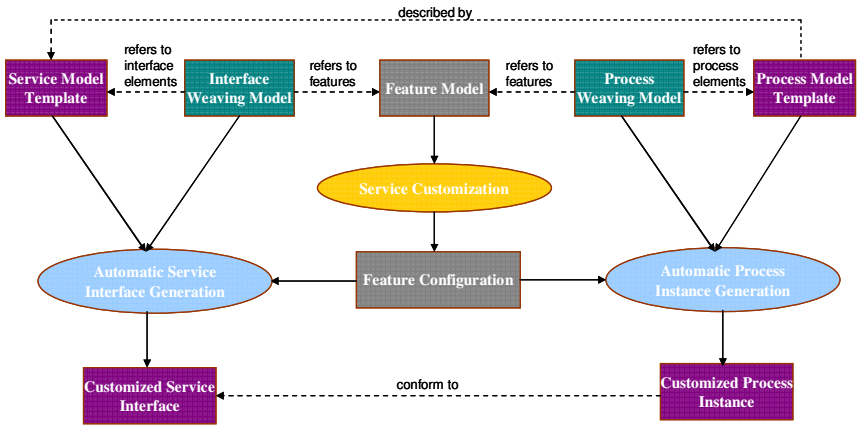

Figure 3. Engineering Process for Customizable Services