Service Customization by Variability Modeling

Michael Stollberg and Marcel Muth

SAP Research
CEC Dresden, Germany
{michael.stollberg,marcel.muth}@sap.com

Abstract. The establishment of service orientation in industry determines the need for efficient engineering technologies that properly support the whole life cycle of service provision and consumption. One challenge is adequate support for service consumers for employing complex services in their individual application context, which becomes particularly important for large-scale enterprise technologies where generic services are designed for reuse in several business scenarios. This paper presents an approach for service customization by model-driven variability management. The variable aspects of the services are explicitly described on the basis of a metamodel. Upon this, service consumers can easily create personalized service variants that properly suit their specific context while the consistency for service invocation is maintained.

1 Introduction

In the last years, service orientation has become the dominating design principle for modern ICT technologies in industry as well as in the public sector. The aim is to exploit the enormous potential of services for enhancing the interoperability among systems and the reuse of implementations. In consequence, a steadily growing number of available services and service-based applications can be observed. The design, development, usage, and management of such solutions requires sophisticated engineering technologies that support the life cycles of service provision and service consumption in an efficient and integrated manner.

This is subject to the emerging discipline of service engineering. Numerous efforts in academia and industry have developed a wealth of techniques, methodologies, and tool support for this. However, existing solutions focus mainly on support for service providers, i.e. for the design, development, publication, and management of services. The consumption side – i.e. the support for service consumers for finding suitable services and integrating them into the specific target application – is often neglected. Existing technology support for this is mostly limited to low-level technical details, leaving the major part of the analysis and integration task for actually consuming services to manual inspection.

The limitations become obvious when considering the consumption of more complex services that commonly occur in real-world business applications. For example, consider the Enterprise Services that form the basis of SAP's modern service-based enterprise technology. These are designed in a generic manner and

A. Dan, F. Gittler, and F. Toumani (Eds.): ICSOC/ServiceWave 2009, LNCS 6275, pp. 425–434, 2010.

cover several usage options, therewith becoming reusable in various business scenarios. On the other hand, their interfaces and usage conditions are considerably complex. Typically, a customer merely requires a subset or a specific flavor of the provided features. Hence, the Enterprise Services need to be configured and integrated in order to properly fit the customer's needs. This is a non-trivial task that requires both technical knowledge and business expertise. Due to the limited tool support, the customization requires massive human involvement and thus becomes a highly cost-intensive and error-prone task.

To overcome this, we present an approach for service customization by the creation of simplified variants that only expose those features that are relevant for the usage scenario of an individual consumer. These are defined on the basis of variability specification models that explicitly describe the usage conditions and constraints of the original service. To support model-driven engineering, we define a metamodel for describing the variable aspects of services, i.e. the mandatory and optional operations, properties of message types as well as their dependencies. We define an engineering process for service customization and provide tool support for specifying service variability models as well as for creating service variants via intuitive user interfaces. Finally, a technical interface is generated from the service variant model. This usually is significantly less complex than the interface of the original service while the consistency for the correct invocation is maintained. Furthermore, the simplified service description can serve as the basis for other service engineering techniques, e.g. for mash-up techniques that are mostly limited to services of limited complexity.

The paper is structured as follows. At first, Section 2 provides a concise overview of the approach and defines the engineering process for service customization. Section 3 specifies the metamodel for service variability modeling, and Section 4 defines the tool-supported procedures for service customization. Section 5 illustrates the techniques and tools for customizing an Enterprise Service, and Section 6 concludes the paper and outlines future work. An extended technical report with a detailed positioning in related work is provided in [5].

2 Overview

The following provides an overview of the approach for service customization. We define the central artifacts and roles involved in the process of providing and consuming customized services, outline the technical solution that is presented in detail in this paper, and motivate the need for such technologies.

Figure 1 provides a comprehensive overview, identifying the involved roles, phases, and relevant artifacts for the engineering process of providing, preparing, and consuming customized services. Abstracting from the concrete formation in industrial service engineering, we distinguish three main roles: the *Service Provider* develops and publishes services, the *Domain Expert* prepares them for customization by defining the variability specification model along with preconfigurations for respective user groups, and the *Service Consumer* customizes and personalizes the service in order to fit it into the specific application context.

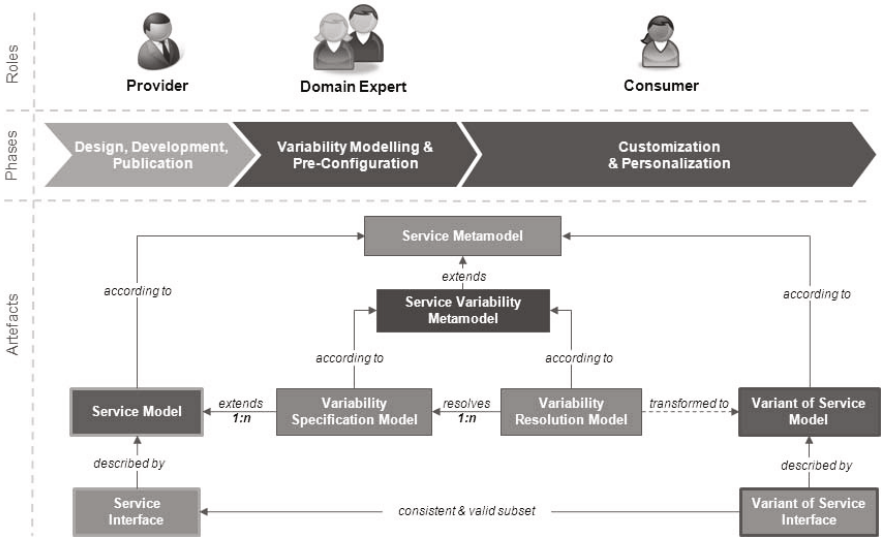

Fig. 1. Service Customization – Roles, Phases, Artifacts

In the first phase, the Service Provider develops a service and publishes it in a repository. In the context of model-driven engineering, the service interface that defines the operations, messages, and endpoints is described by a Service Model on the basis of a metamodel (e.g. a WSDL metamodel or SoaML).In the second phase, the Domain Expert prepares the service for customization. For this, he creates a *Variability Specification Model* that describes the variable aspects of the service (i.e. the mandatory and optional operations, messages, and message types as well as dependencies). This is defined in accordance to the *Service Variability Metamodel* that defines the necessary constructs for modeling the variability of services. The Domain Expert might define multiple variability specification models for a service where each one is pre-configured for a particular application scenario (e.g. for specific industry sectors or geographical usage contexts). In the third phase, the Service Consumer adapts the service to the individual consumption context by defining a *Service Resolution Model*. For this, the variable aspects defined in the Variability Specification Model are resolved by selecting the desired features and by defining concrete values for the parameters that are not changed dynamically during the invocation. Finally, a Service Interface for the variant is generated: this only contains the selected features and represents a valid subset of the original service interface; the explicit variability modeling and the validation of the usage conditions throughout the customization process ensure the correct invocation of the service.

The technical solution for supporting service customization presented in this paper encompasses the specification of the Service Variability Metamodel and tools for supporting the creation of Variability Specification Models by Domain Experts as well as for Variability Resolution by Service Consumers. The overall

idea for enabling service customization by variability modeling is adopted from works on variability management in Software Product Line Engineering (SPLE, [1]), which however deal with different elements and thus employ different models and techniques. In order to ensure the efficiency of the service engineering process, we consider a model-driven approach where the service and variability models are defined on the architecture level (i.e. on the PIM level in terms of MDA [3]). Our prototype implementation works on SoaML [2] as the service metamodel; however, our Service Variability Metamodel is defined orthogonally to the base model, so that it can also be applied to other service metamodels.

The motivation and business relevance of such a service customization technology results from the growing complexity of services, which particularly arises in the context of business applications. For illustration, consider a business service for creating and managing sales orders. A sales order is a relatively complex data object and, furthermore, its detailed structure is defined differently within the standards for different industries. In order to be reusable in various applications within several industries, a general purpose business service needs to support all options and specific features that are relevant for the targeted usage industries. In consequence, its interface becomes very complex regarding the number operations, the size of input- and output objects, and the conditions and constraints for proper and consistent consumption. A specific consumer typically only requires a subset of the provided features for his individual sales order management. However, due to the complexity, the general purpose service is not easy to understand, and its configuration for the individual needs of the consumer is a time consuming task that usually requires external support.

Our approach enables a step-wise reduction of the complexity and improves the technology support for customization. At first, a Domain Expert can define variants that are pre-configured for specific user groups, e.g. one variant for the automotive industry, one for steel production, and another one for the telecommunication sector. Furthermore, the detailed usage conditions for each variant are described explicitly in terms of a variability specification model. On this basis, a consumer can then define a personalized variant by selecting the desired features. The model-driven approach facilitates the abstraction from technical details as well as the provision of intuitive graphical user interfaces for modeling support, and the variability models ensure that the selections by the consumer are compliant with the usage conditions of the service. The generated technical interface for the consumer's variant is naturally significantly less complex than the one of the original sales-order service while it adheres to its usage conditions, so that a correct and consistent invocation is ensured.

Regarding related work, only a works address service customization on higher levels than technical configuration of which most merely provide methodological guidelines but no automated support. For this, our approach adopts concepts of variability modeling from SPLE (s.a.). Some recent works take a similar approach, but rather focus on architectural aspects than on variability management techniques for services for which our approach presents a complementary technique. We refer to [5] for a detailed positioning in related work.

3 Service Variability Metamodel

This section presents the metamodel for describing the variability of services. We first introduce the main elements, and then the constructs for variability modeling on different levels of service descriptions.

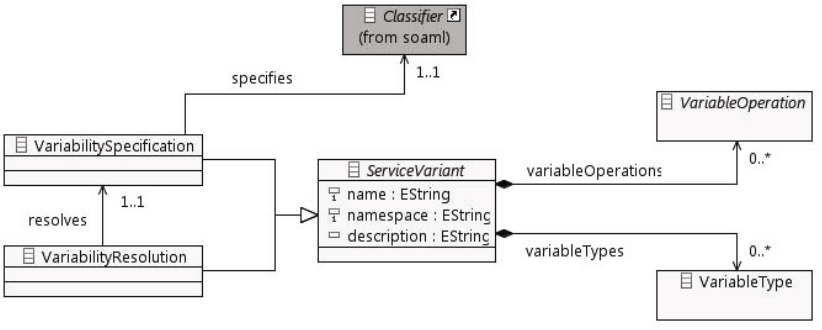

Fig. 2. Service Variability Metamodel – Main Elements

As shown in Figure 2, a *VariabilitySpecification* and *VariabilityResolution* serve as containers for the variability descriptions of the variable artefacts (operations and datatypes) of a specific service (which here is described by a SoaML Classifier [2]). The *VariabilitySpecification* contains the definition of all the variable aspects which are resolved by instances of a *ResolutionElement*. The variability description of a service are modeled with the help of four mechanisms shown at Figure 3:

1. Declaration of mandatory and optional elements, modeled by the boolean `required` property of *VariableElement* which serves as the the superclass for variability modeling on different levels of service descriptions.
2. Definition of dependencies among elements, modelled by the *Constraint* class with direct support for defining excluding and requiring constraints
3. Selection of desired features for a specific application context, modeled by the boolean `selected` property of *ResolutionElement*, and
4. Definition of fixed values that are changed within the application scenario or default values that are used for invocation when no concrete value is provided for, which is only applicable for *VariableProperty* and *PropertyResolutionElement*.

These central constructs are used to describe and handle the variability of services on various aspects. Currently, the metamodel covers the *operation level* where specialized variable elements for operations and their messages are defined and the *data level* where the variability of messages types can be explicitly defined on the level of properties. Due to space limitations, we refer to the extended technical report for further details on this [5].

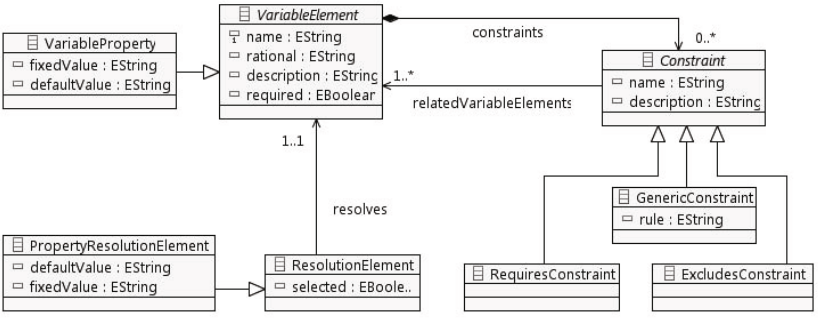

Fig. 3. Service Variability Metamodel – Variability Mechanisms

4 Service Customization Procedures and Tool Support

This section explains the techniques for service customization that work on the metamodel presented above, refining the overall engineering process introduced in Section 2. We here focus on the methodological aspects, while presenting the tooling support in the context of the illustrative example below in Section 5.

4.1 Service Variability Specification

As outlined above, the first step for enabling the customization of services is the creation of a variability specification model. This is performed by a domain expert in the second phase of the overall engineering process (*cf.* Figure 1 above). For this, the domain expert analyses the description of the base service, and creates a variability specifcation along with pre-configurations for the foreseen application context.

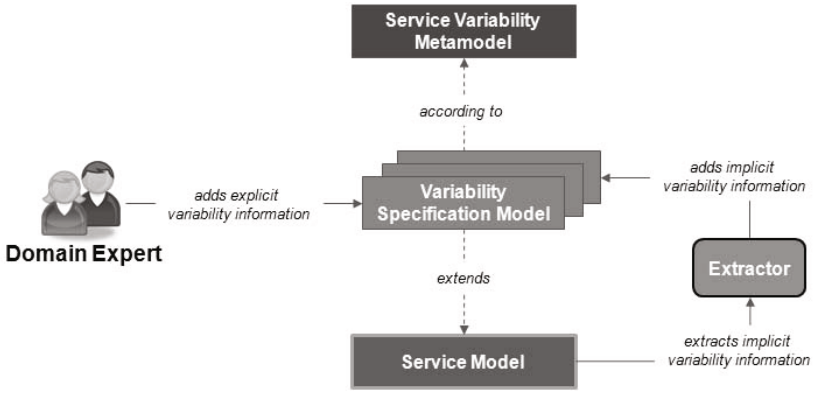

Fig. 4. Variability Specification Modeling

Figure 4 shows the tool-supprted procedure for this. At first, the *Extractor* generates a skeleton of the variability specification model from the original service description, reducing the manual modeling effort. Then, the domain expert can refine the skeleton by adding further variability information, using the mechanisms supported by our metamodel. Note that there can be several variability specification models for a service where each one is pre-configured for a specific application scenario. The distinct variability specification models may define different usage conditions and mandatory and optional fields, as e.g. one industry standard requires fields that are irrelevant in another standard.

4.2 Service Variability Resolution

In the last phase, the consumer creates a service variant that suits the individual application context. For this, he chooses a suitable variability specification model of the service that has been previously prepared by a domain expert, resolve this by selecting the desired features and setting predefined values. From this the consumer generates a variant of the original service model which describes the interface for invoking the service.

Figure 5 shows the detailed procedure for this, which is supported by a graphical engineering tool that we shall present below. A variant is defined by selecting the desired features and defining default and fixed values for type properties and message parameters that will remain unchanged during the actual service invocation. The tool ensures that the user inputs comply with the conditions defined in the variability specification model. Finally, a variant of the original service model is generated from the resolution model. This is described in terms of a conventional service model, and thus can be used invoking the service.

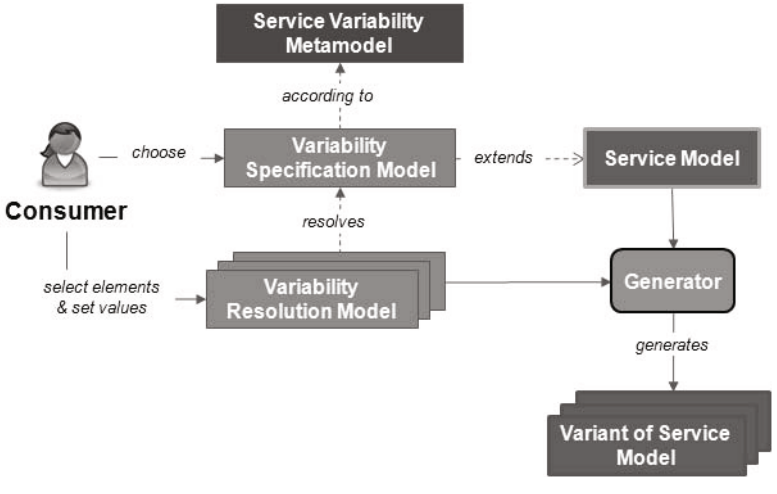

Fig. 5. Service Variability Resolution

5 Illustrative Example

This section illustrates the modeling techniques and procedures introduced above for customizing an SAP Enterprise Service, using our prototype which is implemented as a set of plugins based on the Eclipse Modeling Framework (EMF, see `www.eclipse.org/modeling/emf/`).

5.1 Customizing the Goods Movement Enterprise Service

The Enterprise Service "Goods Movement" provides basic business facilities for managing the movement of goods, offering various usage options that work with complex data structures. For illustration, we will create a variant of the service that only contains the operations and necessary message types for the simple creation and reading of goods movement objects.Our prototype uses SoaML as a platform independent modeling language for the basic service descriptions [2]. Sufficient for demonstration purposes, we here work with already simplified data structures for the message types; the actual business objects used within SAP applications contain more than 100 nodes.

5.2 Variability Specification Modeling

As explained above, the first step in the service customization process is the creation of the variability specification model. Figure 6 shows our prototype for supporting domain experts in this task. This provides an editing facility for variability specification modeling with a tree-view representation and context-sensitive editing support for defining variability modeling elements, constraints, and bindings to the base model.

For the example, the domain expert defines the following explicit variability information. At first, the two operations for creating goods movement objects are grouped into a *ComplexVariableOperation*: conceptually, they present two

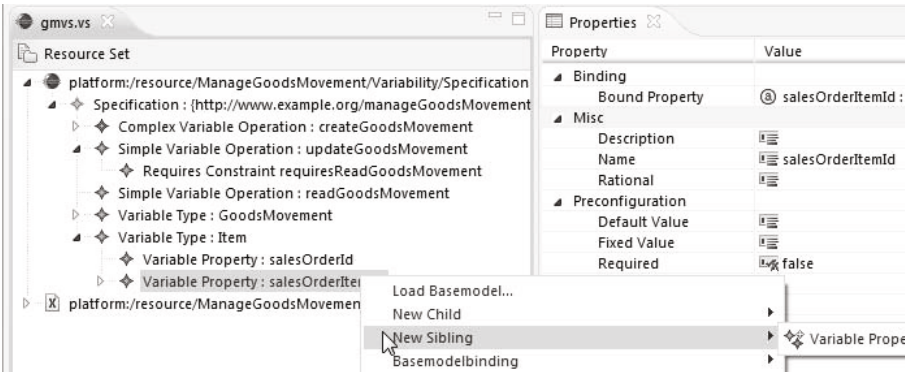

Fig. 6. Variability Specification Tool (Screenshot)

versions of the same operation that differ in the input- and output parameters; however, they are non-exclusive, thus the `multiple`-property is set to `false`. Secondly, the *update*-operation requires the *read*-operation; this is modeled by a `RequiresConstraint` (*cf.* Figure 2). Thirdly, the mandatory elements and their dependencies on the data level are defined. For instance, the usage of *salesOrderItem* is an attribute of the top-level type *item* and requires the *salesOrderId*. In addition, default values can be defined, e.g. setting country information to 'Germany' for pre-configuring the service variant for German customers.

5.3 Variability Resolution Tool

We now turn to the creation of the actual service variant. As explained above in Section 4.2, for this the consumer selects the desired operations and data elements, and defines default or fixed values for static data elements. From this, a conventional service model is generated for the personalized variant. To support consumers in this task, we provide a tool for selecting the desired features via a graphical user interface along with real-time validation of the dependencies and usage conditions defined in the variability specification model.

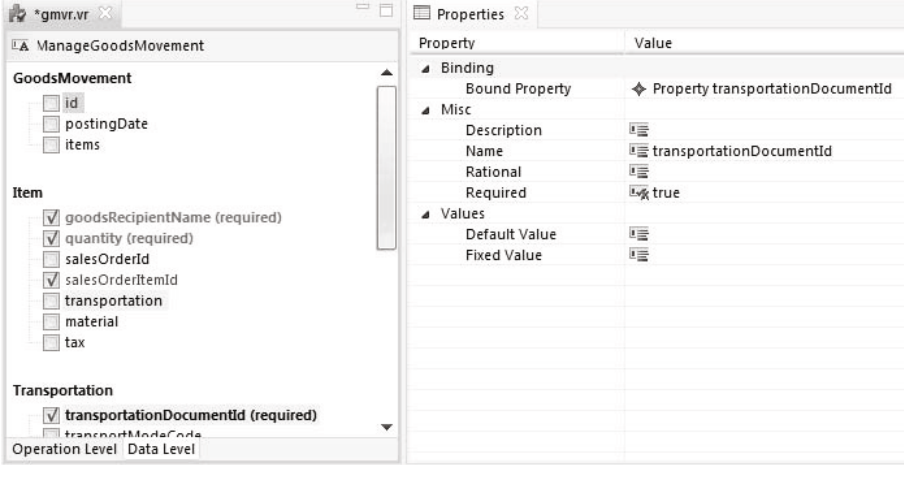

Fig. 7. Variability Resolution Tool (Screenshot)

Figure 7 shows our easy-to-use prototype tool for this. It is organized by tabs for variability resolution on different levels (currently operation and data level), and supports the user by standard metaphors for graphical user interface design [4]: checkboxes for selecting the desired elements, graphical accentuation of required elements, and colored display of constraint violations. The SoaML descriptions for the service variant is generated automatically, and stored in the respective folder of the EMF engineering project.

6 Conclusions and Future Work

This paper has presented an approach for service customization by the tool-supported creation of personalized variants on the basis of variability models that explicitly describe the variable aspects and usage conditions of services.

The need for such techniques arises from the growing number of complex services that are designed for reuse in various application contexts and deal with extensive data objects, which can be particularly found in service-oriented business applications. In order to facilitate the efficient and consumer-oriented consumption, we have proposed three-phased engineering process: domain experts prepare the services published by a provider for specific usage contexts by defining variability specification models; these explicitly describe the variable aspects, upon which consumers can easily create personalized variants that adopt the services to the specific context of the individual application scenarios.

In order to support model-driven development, we have defined a metamodel for describing the variability modeling for services on the basis of four mechanisms: the declaration of mandatory and optional elements with their dependencies, furthermore the selection of desired features as well as the definition of default and fixed values for a particular application scenario. We have developed tools for supporting domain experts in the specification of the service variability as well as consumers in the creation of individualized variants, and we have demonstrated them for customizing an Enterprise Service.

The presented technique enables the consumption of complex services in specific application scenarios. Its main benefits are the minimal modeling effort and ensuring a valid service invocation by the explicit variability modeling and thorough resolution validation. We however consider the presented solution as an initial prototype that shall be extended with additional aspects and further developed towards a comprehensive service customization technology in the future.

Acknowledgements. This paper is mainly based on works supported by EU funding under the SHAPE project (FP7 - 216408).

References

1. Bayer, J., Gerard, S., Haugen, Ø., Mansell, J.X., Møller-Pedersen, B., Oldevik, J., Tessier, P., Thibault, J.-P., Widen, T.: Consolidated Product Line Variability Modeling. In: Käkölä, T., Dueñas, J.C. (eds.) Software Product Lines - Research Issues in Engineering and Management, pp. 195–241. Springer, Heidelberg (2006)
2. Berre, A. (ed.): Service oriented architecture Modeling Language (SoaML) – Specification for the UML Profile and Metamodel for Services (UPMS). Revised submission, OMG, 2008. OMG document: ad/2008-11-01
3. Frankel, D.S.: Model Driven Architecture. Applying MDA to Enterprise Computing. John Wiley & Sons, Chichester (2003)
4. Galitz, W.O.: The Essential Guide to User Interface Design – An Introduction to GUI Design Principles and Techniques, 3rd edn. Wiley, Chichester (2007)
5. Stollberg, M., Muth, M.: Service Customization by Variability Modeling. Extended Technical Report (2010), `http://www.michael-stollberg.de`